



E.PIPHANY E.4 INSTALLATION GUIDE

RELEASE 4.0

JUNE 1999

E.piphany Confidential

E.PIPHANY E.4 INSTALLATION GUIDE

RELEASE 4.0

JUNE 1999

COPYRIGHT AND TRADEMARKS

Copyright © 1997-1999 by E.piphany, Inc. All rights reserved.

This document and the software it describes are furnished under license and may be used or copied only in accordance with such license. Except as permitted by such license, the contents of this document may not be disclosed to third parties, copied, or duplicated in any form, in whole or in part, without the prior written permission of E.piphany, Inc.

This document contains propriety and confidential information of E.piphany, Inc. The contents of this document are for informational use only, and the contents are subject to change without notice. E.piphany, Inc. assumes no responsibility or liability for any errors or inaccuracies that may appear in this document.

Unpublished rights reserved under the copyright Laws of the United States.

E.piphany™ and E.piphany e.4™ are trademarks of E.piphany, Inc. All other products or name brands are trademarks of their respective holders.

Printed in the USA

RESTRICTED RIGHTS LEGEND

Software and accompanying materials acquired with United States Federal Government funds or intended for use within or for any United States federal agency are provided with "Restricted Rights" as defined in DFARS 252.227-7013(c)(1(ii)) or FAR 52.227-19.

TABLE OF CONTENTS

| | |
|-----------------------------------------------------------|------|
| INTRODUCTION | vii |
| About E.piphany e.4 | vii |
| New Features | viii |
| New Features for End Users | viii |
| Enhancements for Administrators | ix |
| Supported Platforms | x |
| About This Guide | x |
| About e.4 Documentation | xi |
| CHAPTER 1: PREPARING TO INSTALL E.PIPHANY SOFTWARE | 13 |
| Order of Installation | 15 |
| Installing and Configuring Your EpiCenter Host Computer | 15 |
| Installing and Configuring a Windows NT Server Host | 16 |
| Hardware Requirements for Windows NT Server | 16 |
| Windows NT Server 4.0 Installation | 18 |
| Disk-Volume Configuration | 19 |
| Network Connectivity | 20 |
| Installing and Configuring a Solaris Host | 20 |
| Hardware Requirements for Solaris | 21 |
| Solaris Installation | 23 |
| Configuration of Solaris Resources | 23 |
| Disk-Volume Configuration | 25 |
| Creating User and Group IDs | 25 |

CONTENTS

| | |
|--------------------------------------------------------------|----|
| Creating Mount Points for Oracle Tablespaces | 25 |
| Network Connectivity | 29 |
| Installing and Configuring Your Database Server | 30 |
| Installing and Configuring SQL Server | 30 |
| Installing SQL Server | 30 |
| Configuring SQL Server | 31 |
| Assigning Data Stores in SQL Server | 32 |
| Installing and Configuring Oracle | 34 |
| Installing or Upgrading Oracle | 34 |
| Configuring Oracle | 38 |
| Creating Control Files | 41 |
| Assigning Data Stores in Oracle | 43 |
| Creating EpiMart and EpiMeta Users | 44 |
| Preparing Your Application Host | 45 |
| Installing Windows NT Server 4.0 on Your Application Host .. | 45 |
| Installing Internet Information Server, Version 4.0 | 45 |
| Installing Microsoft Office Components | 46 |
| Installing and Configuring Connectivity Packages for Your | |
| Database Client Software | 46 |
| SQL Utilities | 47 |
| Oracle8 Utilities | 47 |
| CHAPTER 2: INSTALLING E.PIPHANY SOFTWARE | 49 |
| Performing the Installation | 51 |
| Application Server | 52 |
| Remote Administration | 55 |
| Configuring E.piphany Software | 56 |
| Configuring the E.piphany Web-server Proxy | 56 |
| Setting Up NT Performance Monitoring for Extraction | 57 |
| Verifying Your Installation | 58 |

| | |
|--------------------------------------|----|
| CHAPTER 3: PREPARING TO MIGRATE FROM | |
| PREVIOUS RELEASES | 59 |
| Migrating an EpiMart Database | 59 |
| SQL Server | 60 |
| Oracle | 61 |
| Exporting Metadata | 61 |
| Modifying Extraction Jobs | 62 |

CONTENTS

ABOUT E.PIPHANY

E.piphany is the industry leader in providing analytic applications that operate with Web-based efficiency and ease of use. Using the E.piphany e.4 System, you can integrate back-office and front-office solutions with a diverse range of third-party data sources. These sources include Enterprise Resource Planning (ERP), Customer Relationship Management (CRM), legacy, e-commerce, and front-office applications, along with a variety of external data providers. Because it has been built using open industry standards, the E.piphany e.4 System not only combines all relevant information, it also can interface with customer touchpoint systems such as Web, e-mail, sales-force automation, and call centers.

The E.piphany e.4 System provides quick access to your company's hard-earned store of information, which promotes stronger relationships with your most valuable customers. No matter which applications you have selected, E.piphany can help you build a 1-to-1 enterprise that serves the individual needs of your customers and delivers speed, flexibility and convenience for your day-to-day operations.

ABOUT E.PIPHANY e.4

The E.piphany e.4 System includes a sophisticated datamart, a set of advanced Web-based applications, and a highly configurable user interface. Using this integrated system, technical professionals can deploy customized and polished solutions for a variety of business problems.

The E.piphany datamart includes powerful and flexible data-management tools, including:

- Customizable extractors for a wide variety of data sources
- Built-in data-transformation semantics
- Advanced schema and metadata management
- Powerful performance-acceleration technologies

In addition, E.piphany provides a growing number of packaged solutions. With a minimum of customization, these solutions can help users solve a variety of business challenges using campaign management, on-line analytical processing (OLAP), data mining, and other activities that require integrated data from throughout your enterprise.

E.piphany solutions are easy to maintain and extend, allowing technical professionals greater scope and creativity in providing needed services.

NEW FEATURES

This release contains a number of new features that make E.piphany e.4 applications especially useful and easy to maintain.

NEW FEATURES FOR END USERS

- The Home page is now organized by topics, with links to your favorite reports. You can also display your favorite charts on the Home page.
- New navigation features include a global navigation bar and context-sensitive menus. The navigation bar allows you to find the topic you need. The context-sensitive links and numbered guidance icons that appear in each Web page, allow you to find the answers to your specific questions. Results automatically appear in the same window from which you make a request.

- New clustering, modeling and scoring applications allow you to create predictive models and use them to score lists and identify segments within those lists. For example, you can rank campaign prospects based on their likelihood to purchase. You can score lists by model or by measure, and create lift charts and profitability charts that help you project the potential returns from a campaign.
- A full-featured Campaign Manager is built into E.piphany e.4. You can define treatments, divide your list of potential recipients into segments, create control groups, and assign treatments to cells within each segment. You can download your list of recipients into output files by according to treatment. You can schedule a campaign job to run as soon as possible, on a later date, or on a repeating basis. To close the loop, you can feed the execution results back into the datamart for post-campaign analysis.
- There are no more “Cut and Paste” buttons. Filter settings and other parameters are carried forward implicitly and seamlessly from one Web page to another. For example, when you carry your list definition into an analysis report, the list is automatically converted into a filter.

ENHANCEMENTS FOR ADMINISTRATORS

- Incremental Nightly Processing delivers valuable efficiency and scalability. Rather than processing semantics and aggregates on all the data, you can limit operations to the increments that have changed. The time savings can be substantial.
- Sharable Web-page contents simplify much of your work. You can set up the filters, options, attributes, and measures (FOAM) just the way you want them, and then reuse those elements in a variety of Web pages.
- The Aggregate Optimizer helps you to identify the aggregates that will give your users the best response-times. You can find out which aggregates are being used, and which ones you can drop. You can also identify the aggregate that best covers a frequent user query and check its size before you add it.

- A growing list of packaged applications allow you to deploy new solutions quickly.

SUPPORTED PLATFORMS

This release of E.piphany e.4 supports the implementation of an EpiCenter datamart on the following database-server platforms:

- Oracle8 on Solaris 2.6 (Version 8.0.5.1)
- SQL Server 7.0, Enterprise Edition, on Windows NT Server 4.0 Enterprise Edition (with Option Pack 4 and the latest service pack)

For further information about this release of E.piphany e.4 software, refer to the Release Notes. To review the Release Notes after you have installed your E.piphany e.4 system, click the Microsoft Windows **Start** button, then choose **Programs**, then **Epiphany**, then the name of your Release 4.0 instance, then **Documentation**, and then **Release Notes**.

ABOUT THIS GUIDE

This guide is intended for database administrators and consultants who install, configure, and maintain E.piphany datamarts and solutions. Please follow these guidelines when you install Epiphany software.

1. Read Chapter 1 of this guide to ensure that you understand the following topics:
 - The considerations involved in preparing the host computer, the operating system (OS), the database server, and networking connectivity for your EpiCenter datamart
 - The steps that are required to prepare the host computer for your E.piphany application software

2. Make sure that you have the appropriate installation and configuration manuals on hand for the hardware and software products on which your Epiphany software depends. Epiphany suggests that you obtain the following manuals, depending upon the database-server option you select.
 - *Net8 Getting Started* (for use with Windows NT Server clients connected to Oracle/Solaris datamarts)
 - *Oracle8 Installation Guide for SunSPARC Solaris*
 - *Oracle8 Reference*
 - *Solaris 2.6 Hardware, SMCC Hardware Platform Guide*
 - *Solaris 2.6 SPARC Installation Instructions*
 - *SQL Server Administration Guide* (for Windows NT Server)
 - *Start Here: Basics and Installation, Microsoft Windows NT Server*
3. Please install the hardware and software products that are mentioned in this guide according to manufacturer instructions.
4. Please follow the configuration recommendations that are suggested in this guide.

If specific instructions do not appear for a particular configuration step or option, default values are acceptable.

ABOUT e.4 DOCUMENTATION

All of the manuals in the Epiphany e.4 documentation library are available for viewing on-line with Adobe Acrobat Reader. To view a manual in the documentation set, choose **Programs** from the Microsoft Windows **Start** menu, then **Epiphany**, then the name of your Release 4.0 instance, then **Documentation**, and then the title of your choice. To review additions and corrections that were inadvertently omitted from a manual, choose **Updates**.

Epiphany provides a full-text search index that allows you to perform keyword searches across all of the manuals in the documentation set. You must have Acrobat Reader with Search[®] or Acrobat Exchange[®] installed on your computer to use the index. To obtain a free copy of Acrobat Reader with Search, visit the following Web site:

<http://www.adobe.com/prodindex/acrobat/readstep.html>

To perform a keyword search in Acrobat Reader with Search or Acrobat Exchange:

1. Choose **Search** and then **Query** from the **Tools** menu
2. Enter a keyword or phrase to search for in the Keyword Search dialog box
3. Select the appropriate search options.
4. Click **Search**.
5. Choose a document in the Search Results dialog box.

Acrobat Exchange and Acrobat Reader with Search include tools that enable you to expand and limit your search criteria. For complete instructions on using the search capabilities of Adobe Acrobat, choose **Search Online Guide** from the **Help** menu.

PREPARING TO INSTALL E.PIPHANY SOFTWARE

The process of installing E.piphany software requires that you first install and configure hardware and software components that are supplied by other manufacturers. The success of your E.piphany application depends on the correct installation and proper configuration of these components.

This manual provides general installation guidelines and configuration recommendations for the products on which E.piphany software depends. For specific instructions about a particular product, please refer to the manufacturer's documentation. The configuration recommendations discussed in this manual apply to E.piphany enterprise-relationship-management (ERM) applications only.

The components that you must install and configure before installing E.piphany software include:

- A dedicated host computer for your EpiCenter datamart (strongly recommended) running one of the following operating systems:
 - Windows NT Server 4.0, Enterprise Edition with Service Pack 4 and the Microsoft Exchange mail-client utility from Options Pack 4
 - Sparc Solaris 2.6
- RAID disk-volume support (optional but recommended)
- A relational database server from the following list:
 - SQL Server 7.0 (E.piphany recommends the Enterprise Edition)
 - Oracle8, Release 8.0.5.1.0 for Solaris, with the Oracle Partitioning Option

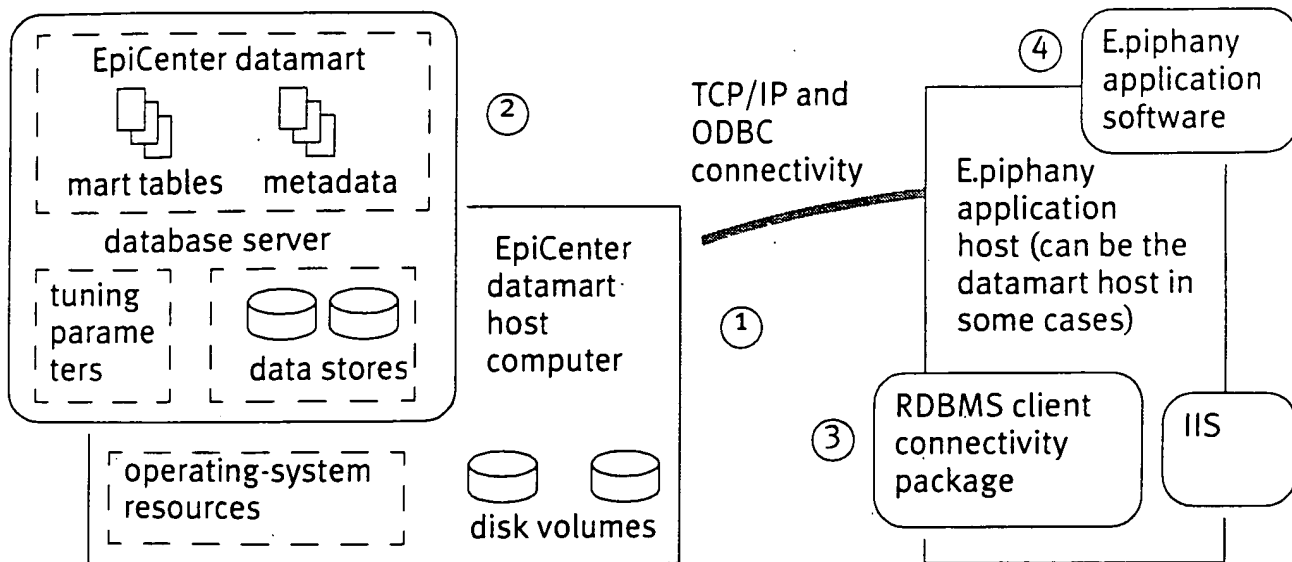
- A host computer for your E.piphany application running Windows NT Server
- Client utilities for your application host, including:
 - Microsoft Internet Information Server (IIS), Version 4.0, and the Microsoft Exchange Mail utility (available with Option Pack 4 for Windows NT Server)
 - The appropriate connectivity and management package:
 - SQL Server Utilities with a datamart that resides on SQL Server
 - Oracle8 Client for Windows with a datamart that resides on Oracle

Note: A separate application host is required only when you install your EpiCenter datamart on a Solaris host. If you install your datamart and E.piphany application software on the same computer, you must install the appropriate client utilities listed above on your datamart host.

ORDER OF INSTALLATION

Figure 1 illustrates the order in which you install and configure the components of your E.piphany application.

FIGURE 1: EPIPHANY INSTALLATION COMPONENTS



INSTALLING AND CONFIGURING YOUR EPICENTER HOST COMPUTER

The specific installation and configuration tasks you must perform depend upon the host computer on which you intend to install your EpiCenter datamart. The sections that follow discuss the installation and configuration process for the following hosts:

- Pentium-based computers running Windows NT Server 4.0
- Sparc-based computers running Solaris 2.6

INSTALLING AND CONFIGURING A WINDOWS NT SERVER HOST

This section provides guidelines for installing and configuring a computer running Windows NT Server 4.0 as an EpiCenter host. See “Installing and Configuring a Solaris Host,” on page 20 for information about Solaris hosts.

HARDWARE REQUIREMENTS FOR WINDOWS NT SERVER

Please ensure that your computer meets the following minimum requirements:

- At least one CPU with a speed of 200 megahertz or higher (two or more CPUs are preferred), capable of supporting Windows NT Server 4.0, Enterprise Edition

Refer to the Microsoft manual entitled *Hardware Compatibility List; Microsoft Windows NT*, Version 4.0, for listings of computers that support Windows NT Server.

- The full complement of random-access memory (RAM) that your computer supports (no less than 128 megabytes)
- Depending on the amount of physical memory that you have installed, space of not less than two times the size of RAM
- A TCP/IP network controller installed and connected to a local area network (LAN)

Multiple network-card configurations are not currently supported.

- An adequate amount of disk space:

Epiphany software requires about 30 megabytes of disk space (including supporting software such as ODBC and JDBC drivers). This amount is in addition to the size of your datamart, which depends on the amount of data that you extract from your source systems. If you already have specifications for your datamart, you can use that information to estimate the size of your datamart data, taking into account the following considerations:

- The datamart includes two copies of the database and aggregates that are between three and ten times the base-table size, with additional space needed for staging tables, indexes, keys, and 100 megabytes of E.piphany metadata.
- E.piphany recommends that you reserve a separate log volume that is about 10 percent of the size of your datamart.
- E.piphany recommends that you reserve a volume that is at least twice the size of your largest fact table for temporary tables.

Based on these considerations and depending on the data storage hardware that your computer is equipped with, E.piphany suggests that you reserve the following disk volumes for your EpiCenter datamart:

- a RAID-1+0 volume for your EpiMart data and EpiMeta metadata
Use the following formula to determine the size of this volume:

$$\text{mart_vol_size} = 2 * (\text{aggs} * \text{fact_rows} * (100 + \text{fact_row_width})) + 2 * (\text{dimension_rows} * (20 + \text{dimension_row_width})) + 400 \text{ Mb}$$

Replace *aggs* with a value between 3 and 10 depending on the degree to which your application requires aggregate data. Use a higher value for applications that require extensive use of aggregate data.

- a RAID-1 volume for the log device associated with your datamart
 $\text{log_vol_size} = (0.1 * \text{mart_vol_size}) + 10\text{MB}$

- a RAID-5 volume for the temporary database

Use the following formula to calculate the size of this volume:

$$\text{temp_db_size} = \text{tmp_factor} * \text{fact_rows} * (100 + \text{fact_row_width})$$

Replace *tmp_factor* with a value between 2 and 10, depending on your application.

E.piphany recommends that you format all disk volumes as NTFS drives, each with a block size of 64 kilobytes.

Note: The performance and costs of RAID volumes are heavily dependent on the specific hardware that you use. The documentation for your RAID equipment might include recommendations that differ from the suggestions made above. If so, E.piphany recommends that you follow the manufacturer's recommendations.

WINDOWS NT SERVER 4.0 INSTALLATION

You must install Microsoft Windows NT Server 4.0 Enterprise Edition, Service Pack 4 (or higher), and selected Microsoft Office components.

INSTALLING WINDOWS NT SERVER

To install Windows NT Server 4.0, follow the directions provided in Part 2 of the Microsoft manual entitled *Start Here: Basics and Installation, Microsoft Windows NT Server, Version 4.0*, and follow these recommendations:

1. Select the default directory location in which to install Windows NT Server.
2. Assign a computer name of 15 or fewer characters and write this name down for future use.
3. Indicate the server type as a stand-alone server.
4. Enter your administrator account name and password.
5. Create a repair disk in case of an emergency.
6. Select the default list of components.

As part of the Windows NT Setup, after you finish setting up the network parameters, the Finishing Setup dialog box is displayed, which informs you that are about to start setting up Microsoft Internet Information Server, Version 2.0. Exit the installer at this time. You do not need to install IIS, Version 2.

Note: When installing the operating system, you might have to install additional drivers for 3rd-party disk drives, graphics cards, or other hardware.

INSTALLING OPTION PACK 4

Detailed instructions for installing Option Pack 4 are provided on the installation CD-ROM. Please follow those instructions to install Microsoft Internet Information Server, Version 4.x.

INSTALLING SERVICE PACK 4 (OR HIGHER)

Detailed instructions for installing your service pack are provided on the installation CD-ROM. Please follow those instructions to install the service pack.

Note: You must install Option Pack 4 before you install the latest Service Pack.

INSTALLING MICROSOFT OFFICE COMPONENTS

You can install Microsoft Office from the Microsoft Office CD. The installation program provides detailed instructions. Please follow those instructions to install the Microsoft Exchange messaging client, ODBC connection client, and any other components of Microsoft Office that you choose to include.

DISK-VOLUME CONFIGURATION

Take the following steps to configure disk volumes for your datamart:

1. The disk volumes for your database server use NTFS format. Some computer models running Windows NT Server configure disks with FAT format by default. To convert disks from FAT to NTFS, use the Disk Administrator. From the Start Menu, choose Programs, then Administrative Tools, and then Disk Administrator.
2. Refer to the disk-volume size figures that you calculated using the formulas in "Hardware Requirements for Windows NT Server," on page 16 for size and RAID-level information.

3. If you are using a hardware RAID controller or RAID software, follow the manufacturer's instructions for configuring your disk volumes. Otherwise, use the Disk Administrator to configure your disk volumes.
4. In a command-window prompt, enter the following command to enable monitoring of disk I/O performance with the PerfMonitor utility:

```
diskperf -y
```

NETWORK CONNECTIVITY

Epiphany software requires the TCP/IP network protocol. Please contact your network administrator for specific instructions regarding the installation and configuration of devices, drivers, and domains for this protocol.

When you have verified that you have network connectivity, you can proceed to the next stage of preparations. Refer to "Installing and Configuring SQL Server," on page 30 for further instructions.

INSTALLING AND CONFIGURING A SOLARIS HOST

This section provides guidelines for installing and configuring a computer running Solaris 2.6 as an EpiCenter host. Please review this section even if you plan to install your E.piphany datamart on an existing Oracle host.

This section discusses a number of topics involved in installing and configuring Solaris for use with Oracle8. For additional information, refer to the *Oracle8 Installation Guide for SunSPARC Solaris* and *Oracle Support Bulletin 104511.69*.

For information about a Windows NT Server host, see "Installing and Configuring a Windows NT Server Host," on page 16.

HARDWARE REQUIREMENTS FOR SOLARIS

Please ensure that your computer meets the following minimum requirements:

- At least one CPU with a speed of 200 megahertz or higher (two or more CPUs are preferred), capable of supporting Solaris 2.6
- The full complement of random-access memory (RAM) that your computer supports (no less than 128 megabytes)
- Depending on the amount of physical memory that you have installed, space of not less than two times the size of RAM
- A Solaris-supported CD-ROM drive that uses High Sierra or ISO 9660 format with the Rockridge extension
- A TCP/IP interface installed and connected to a local area network (LAN)
- An adequate amount of disk space:

Epiphany software requires about 30 megabytes of disk space (including supporting software such as ODBC and JDBC drivers). This amount is in addition to the size of your datamart, which depends on the amount of data that you extract from your source systems. If you already have specifications for your datamart, you can use that information to estimate the size of your datamart data, taking into account the following considerations:

- The datamart includes two copies of the database and aggregates that are between three and ten times the base-table size, with additional space included for staging tables, indexes, keys, and 100 megabytes of E.piphany metadata.
- E.piphany recommends that you provide space for temporary tables that is at least twice the size of your largest fact table, and space for rollback segments that 10 percent of the size of your datamart.
- E.piphany recommends that you use separate devices for redo logs and archived log files.

Based on these considerations, E.piphany suggests that you use the following formula to calculate the size of your datamart:

```
data_size = 2 * (aggs * fact_rows * (100 + fact_row_width))
            + 2 * (dimension_rows * (20 + dimension_row_width)) + 400MB
            + (0.1 * mart_vol_size) + 10MB
            + tmp * fact_rows * (100 + fact_row_width)

rollback_seg_size =      mart_size / 5

mart_size =      data_size + rollback_seg_size
```

Replace *aggs* with a value between 3 and 10, depending on the degree to which you expect your application to require aggregate data. Use a higher value for applications that require extensive use of aggregate data. Replace *tmp* with a value between 3 and 10, depending on the degree to which you expect your application to require temporary tables.

E.piphany recommends that you configure a RAID 1+0 array as the disk volume for your datamart data. For enhanced performance, you might choose to spread datamart data across two or more such disk volumes, depending on the storage devices and controllers that are available to you.

E.piphany also recommends that you configure the following additional disk volumes:

- A volume for the Oracle installation files, initialization files, and redo logs
- A volume for Oracle system tables and log archives

In keeping with the Optimal Flexible Architecture guidelines for Oracle, E.piphany recommends choosing the following volume names:

- **/u01** for system rollback segments
- **/u02** for system tables
- **/u03** for datamart data

If you configure additional disk volumes for datamart data, E.piphany suggests that you begin the numbering scheme for those volume-name suffixes with 4 (**/u04**).

Note: The performance and costs of RAID volumes are heavily dependent on the specific hardware that you use. The documentation for your RAID equipment might include recommendations that differ from the suggestions made above. If so, E.piphany recommends that you follow the manufacturer's recommendations.

SOLARIS INSTALLATION

Follow the directions provided with your Solaris software to install the latest Solaris operating system and current patch level from Sun Microsystems.

CONFIGURATION OF SOLARIS RESOURCES

You must configure the following Solaris resources to support an Oracle database server for your EpiCenter datamart:

- Semaphores, shared-memory segments, and the paging threshold in the `/etc/system` file
- Disk volumes for Oracle data
- User **oracle**, group **dba**, and user **epichnl**
- Mount points for Oracle tablespaces, redo logs, and log archives
- Automatic start-up and shut-down for Oracle
- Network Connectivity

The sections that follow describe these activities.

UPDATING THE /ETC/SYSTEM FILE

As **root**, you must edit the `/etc/system` file to configure the following system resources for Oracle:

- Semaphores
- Shared memory
- Swapping threshold

Increasing values for these parameters allocates more resources, which reduces the amount of physical memory that is available to processes. Add or modify the following lines in **/etc/system** to configure these resources:

```
set semsys:seminfo_semmni=70
set semsys:seminfo_semmsl=100
set semsys:seminfo_semmns=200

set shmsys:shminfo_shmseg=10
set shmsys:shminfo_shmmni=100
set shmsys:shminfo_shmmin=1
set shmsys:shminfo_shmmax=max_shared_size

set lotsfree=pages
```

1. Update the values for the **semmni**, **semmsl**, **semmns**, **shmseg**, **shmmni** and **shmmin** parameters as indicated above. If you are adding an Oracle instance to a host on which another instance already resides, multiply the values indicated above for each parameter by the number of instances that are to be supported on this host.
2. Add a new line to set the **shmax** resource. Replace *max_shared_size* with a value that is between 80 percent and 90 percent of the size of physical memory in bytes. This parameter indicates a total allowable size for shared memory; it does not allocate any memory resources.
3. Update the value for the **lotsfree** parameter. Replace *pages* with the minimum number of pages that your computer requires in order to prevent unnecessary swapping. E.piphany recommends a value between 2 percent and 6 percent of the total number of pages in RAM. Use the following formula calculate the number of 8-kilobyte pages in RAM:

$$\text{total_pages} = \text{megs} * 128$$

Replace *megs* with the number of megabytes of RAM that are installed on your computer.

4. Use the **reboot** command to reboot your Solaris host so that these semaphore and shared-memory settings can take effect.

DISK-VOLUME CONFIGURATION

As **root**, take the following steps to configure disk volumes for your datamart:

1. Refer to the disk-volume size figures that you calculated using the formulas in “Hardware Requirements for Solaris,” on page 21 for size and RAID-level information.
2. If you are using a hardware RAID controller or RAID software, follow the manufacturer’s instructions for configuring your disk volumes. Otherwise, use the standard Solaris utilities for configuring disk partitions.

CREATING USER AND GROUP IDS

As **root**, take the following steps to create the **oracle** and **epichnl** user IDs and the **dba** group, if those IDs are not already present.

1. Use **admintool** or the **useradd** utility to create the **oracle** and **epichnl** user IDs and the **dba** group.
2. Set the default group ID for both **oracle** and **epichnl** to **dba**.
3. Propagate this account and group information to the Network Information Service (NIS).

CREATING MOUNT POINTS FOR ORACLE TABLESPACES

As **root**, take the following steps to create mount points for Oracle:

1. Use the **mkdir** command to create mount points with names of the following form on the disk volumes that you have reserved for Oracle:

```
mkdir /u01/u02 /u03 /u04
```

These mount points correspond to the Optimal Flexible Architecture (OFA) specification for an Oracle database server. The mount points are used by Oracle as follows:

- **/u01** Oracle system tables
- **/u02** system rollback segments

- /u03 datamart tablespaces
 - /u04 additional datamart tablespaces (optional)
2. Use the following commands to assign user ownership, group ownership, and access permissions to each of these mount points.

```
chown oracle /u*
chgrp dba /u*
chmod 775 /u*
```

MODIFYING THE SHELL INITIALIZATION FILES

Take the following steps to modify the shell initialization files for the **oracle** and **epichnl** user IDs.

1. Log in as user **oracle**.
2. Edit the **.profile** file in the **oracle** home directory to add the following lines, which you can copy from the **/cdrom/resources/profile.sh** file on the E.piphany installation CD:

```
#DISPLAY=hostname:0.0# Uncomment this line and replace hostname
                        # with the name of the
                        # installation host.

ORACLE_BASE=/opt/oracle # Replace with actual pathname.
ORACLE_TERM=$TERM      # (Optional) Replace $TERM with
                        # alternate terminal type for Oracle.

MNT1=/u01              # Oracle system tables
MNT2=/u02              # Oracle system rollback segments
MNT3=/u03              # Datamart tables
MNT4=/u04              # Additional datamart tables (optional)

ORACLE_SID=ORCL        # Replace ORCL with actual SID.
ORACLE_HOME=/ $ORACLE_BASE/product/8.0.5
PATH=.: $ORACLE_HOME/bin: $ORACLE_BASE/local:/bin:/usr/bin:\
/usr/ccs/bin:/usr/openwin/bin:/usr/5bin:/usr/ucb
LD_LIBRARY_PATH=$LD_LIBRARY_PATH: $ORACLE_HOME/lib
TNS_ADMIN=$ORACLE_HOME/network/admin
ORA_NLS33=$ORACLE_HOME/ocommon/nls/admin/data
export ORACLE_BASE ORACLE_SID ORACLE_HOME ORACLE_TERM PATH \
DISPLAY LD_LIBRARY_PATH TNS_ADMIN ORA_NLS33 MNT1 MNT2 MNT3 MNT4
```

```
ORAENV_ASK=no
umask 022
```

3. Make the following edits to the **.profile** file:
 - a) If you are installing from a remote host, uncomment the line that sets the DISPLAY environment variable and replace *hostname* with the name of the remote host.
 - b) If the TERM variable is not set in your environment, or if you want to use a terminal type for Oracle utilities that differs from the default terminal type for your shell, replace \$TERM with the appropriate value for that terminal type.
 - c) Be sure that the pathname for the ORACLE_BASE environment variable is correct.
 - d) Set the ORACLE_SID environment variable to the correct instance name for your EpiCenter instance.
4. If you use the C shell, edit the **.cshrc** file to add the following lines, which you can copy from the **/cdrom/resources/cshrc.csh** file of the installation CD. Perform the same edits to this file that you did in the previous step. EpiPhany recommends that you update the **.cshrc** file in addition to, rather than instead of, the **.profile** file, so that users of any common shell can have environment variables set up automatically.

```
#setenv DISPLAY hostname:0.0# Uncomment this line and replace
# hostname with name of host for
# remote installation.

setenv ORACLE_BASE /opt/oracle # Match pathname in .profile.
ORACLE_TERM=$TERM # (Optional) Replace $TERM with
# alternate terminal type for Oracle.

setenv MNT1 /u01 # Oracle system tables
setenv MNT2 /u02 # Oracle system rollback segments
setenv MNT3 /u03 # Datamart tables
setenv MNT4 /u04 # Additional datamart tables (optional)

setenv ORACLE_SID ORCL# Replace ORCL with the actual SID.
setenv ORACLE_HOME $ORACLE_BASE/product/8.0.5
setenv PATH .:$ORACLE_HOME/bin:$ORACLE_BASE/local:/bin:\
/usr/bin:/usr/ccs/bin:/usr/openwin/bin:/usr/5bin:/usr/ucb
setenv LD_LIBRARY_PATH $LD_LIBRARY_PATH:$ORACLE_HOME/lib
```

```
setenv TNS_ADMIN $ORACLE_HOME/network/admin
setenv ORA_NLS33 $ORACLE_HOME/ocommon/nls/admin/data
set ORAENV_ASK=no
umask 022
```

5. Log in as user **epichnl** and copy the **.profile** and **.cshrc** files that you just edited from the **oracle** home directory to the home directory for **epichnl**.
6. Enable remote shell access for user **epichnl** by adding the following line to the **.rhosts** file in the **epichnl** home directory:

```
application_host          epichnl
```

Replace *application_host* with the hostname of the Windows NT Server host computer on which you intend to install your E.piphany application. Entries in the **.rhosts** file are case sensitive, so be sure that you enter the hostname exactly as it appears in the NT domain.

SETTING UP AUTOMATIC START-UP AND SHUT-DOWN FOR ORACLE

Take the following steps to set up automatic start-up and shut-down for Oracle:

1. Log in as **root**.
2. Enter the following command lines into the **/etc/init.d/dbora** file. You can copy this file from the **/cdrom/resources/dbora.sh** file of the installation CD. Replace *oracle_home_path* with the value of **ORACLE_HOME** as it appears in the **.profile** file that you created in the previous section.

```
ORACLE_HOME=oracle_home_path
ORACLE_OWNER=oracle
if [ ! -f $ORACLE_HOME/bin/dbstart -o ! -d $ORACLE_HOME ]
then
    echo "oracle startup: cannot start"
    exit
fi
case "$1" in
'start')
    su - $ORACLE_OWNER -c $ORACLE_HOME/bin/dbstart &
;;
```

```
'stop')
    su - $ORACLE_OWNER -c $ORACLE_HOME/bin/dbshut &
;;
esac
su - $ORACLE_OWNER -c "lsnrctl start"
```

3. Create links to the **dbora** file in the system-initialization run-level directories:

```
ln -s /etc/init.d/dbora /etc/rc0.d/K10dbora
ln -s /etc/init.d/dbora /etc/rc2.d/S99dbora
```

4. Create a symbolic link for the **listener.ora** file:

```
ln -s /var/opt/oracle/listener.ora /etc/listener.ora
```

NETWORK CONNECTIVITY

The following steps enable network access for the Oracle8 database server:

1. Enable remote shell access from your E.piphany application host by adding an entry of the following form to the DNS hosts domain or the **/etc/hosts** file:

```
IP_address hostname
```

Replace *IP_address* with the IP address of the application host. Replace *hostname* with the hostname of the application host.

2. Add the following entry to the **/etc/services** file for your Oracle database server. If you are adding an Oracle instance to a host that already has one, you must add an additional **listener** entry for the new instance with a distinct port number.

```
listener      port/tcp      #TNS Listener
```

Replace *port* with the port number for your Oracle database server or Net8 service. E.piphany suggests port 1521 if you do not already have that port assigned.

3. Log in as a user other than **root**. Then use the **ftp** command to verify that your Solaris host is connected to the network and can transfer data.

When you have verified that you have network connectivity, you can proceed to the next stage of preparations. See “Installing and Configuring Oracle,” on page 34 for further instructions.

INSTALLING AND CONFIGURING YOUR DATABASE SERVER

This section provides guidelines and recommendations for:

- Installing and configuring the database server for your EpiCenter datamart
- Assigning data stores for your EpiCenter data
- Creating databases for your EpiCenter datamart and metadata

INSTALLING AND CONFIGURING SQL SERVER

This section provides guidelines and recommendations for installing and configuring SQL Server for use with an EpiCenter datamart. You must install the Enterprise Edition of SQL Server Version 7.0. This Edition can be installed only on hosts that are running the Enterprise Edition of Windows NT Server.

INSTALLING SQL SERVER

To install SQL Server, follow the directions provided on the installation CD, along with the following recommendations.

1. Choose Install Prerequisites from the initial installer dialog, then install Internet Explorer 4.01, Service Pack 1.
2. Choose Install SQL Server 7.0 Components from the initial dialog.
3. In the Install SQL Server 7.0 Components dialog, choose Enterprise Edition.
4. In the Select Install Method dialog, choose Local Install.

5. In the Convert Existing SQL Server Data dialog, check Yes, run the SQL Server Upgrade Wizard.
6. In the Setup Type dialog, choose the default values (Typical installation, default destination folders for program files and data files)
7. In the Service Accounts dialog, choose:
 - Use the same account for each service
 - Auto Start SQL Server Service
 - Use the Local System Account
8. If you are asked to choose a character set, choose 850 Multilingual.
9. If you are asked to choose a collating sequence, choose Unicode.
10. If you are asked to choose a sort order, choose Dictionary Order, Case Insensitive.
11. If you are asked to choose networking protocols, include TCP/IP sockets and the default port number.
12. If you are prompted to run the Upgrade Wizard, answer Yes, then follow the instructions in the Upgrade Wizard.

CONFIGURING SQL SERVER

Configuring SQL Server 7.0 involves the following actions:

- Registering and starting the server
- Configuring parallelism

REGISTERING AND STARTING SQL SERVER

To register you server and start SQL Server, take the following steps:

1. From the Start menu, choose Programs, then Microsoft SQL Server, and then Enterprise Manager.
2. Expand the SQL Server Group folder, then right-click the icon for your server.

3. Choose New Server Registration to bring up the Register SQL Server Wizard.
4. In the Select an SQL Server dialog, enter the name of your server in the text box above the list of available servers, then click Add.
5. In the Select Authentication Mode dialog, choose SQL Server authentication.
6. In the Select Server Group dialog, choose SQL Server Group.
7. Continue with this wizard to completion.
8. Expand the SQL Server Group folder to see your new server icon.
9. Right-click the icon for your server, then click Start.

CONFIGURING PARALLELISM

To configure parallelism, take the following steps:

1. Right-click the icon for your server, then choose Properties.
2. In the Server Properties dialog, choose the Processor tab.
3. Check the Boost SQL Server priority on Windows NT box.
4. Do not check the Use Windows NT Threads box.
5. If your host is dedicated exclusively to your datamart and you plan to run your E.piphany application on another computer, click the Use all available processors button. If you plan to run your E.piphany application on this same host, click the **Use ... processor(s)** button. Enter one less than the number of CPUs that reside on your host.

ASSIGNING DATA STORES IN SQL SERVER

Use the SQL Server Manager to set up new database devices and new databases in your EpiCenter datamart. You need to set up the following databases:

- A database for your datamart data

- A database for E.piphany metadata
- A temporary database

You must also create the transaction logs associated with each of these databases. E.piphany recommends that you place transaction logs on different disks than those on which your databases reside.

Take the following steps to set up each database:

1. Expand the folder for your database server in the Enterprise Manager.
2. Right-click on the Databases folder, then choose New Database.
3. Enter the following information in the General tab of the Database Properties dialog:
 - The name of the new database
 - The location

Click the button next to the entry for your new database in the Location column to browse for a location.
 - Automatic file growth in 10-megabyte increments
 - Unrestricted file growth
4. Enter the following information in the Transaction Log tab:
 - The location (click the button and browse)
 - Automatic file growth in 5-megabyte increments
 - Unrestricted file growth
5. Right-click the icon for you new database, then choose Properties.
6. In the Options tab of the Properties dialog, check only the following options. Uncheck any other options that might have been selected by default.
 - Select into / bulk copy
 - Truncate log on checkpoint
 - Auto create statistics

When you have completed these steps for each of the three databases, you can proceed to “Preparing Your Application Host,” on page 45.

INSTALLING AND CONFIGURING ORACLE

This section provides guidelines and recommendations for installing and configuring Oracle8 for use with an EpiCenter datamart. Please review these sections even if you plan to use an existing Oracle instance to house your EpiCenter datamart.

INSTALLING OR UPGRADING ORACLE

The installation process for Oracle takes place in the following stages. Please complete each stage before proceeding to the next one:

- Installing or upgrading Oracle Version 8.0.5.0
- Installing the Oracle 8.0.5.1.0 (or higher) patch release
- Running the **root.sh** script (new installations only)

INSTALLING OR UPGRADING VERSION 8.0.5.0

If you have not already done so, please follow the directions provided in the *Oracle8 Installation Guide* to install Oracle or upgrade your Oracle instance to Version 8.0.5.0. The following steps summarize the installation process for this Oracle release.

1. Log in as user **oracle**.
2. Enter the following command to enable access to the window system on the local host. Replace *hostname* with the name of the local computer.

`xhost hostname`
3. Use one of the following commands to assign the local host as the display device for the Oracle installation GUI:

```
DISPLAY=hostname:0.0 ; export DISPLAY    # for sh or ksh
setenv DISPLAY hostname:0.0             # for csh
```

4. Change directory (**cd**) to **/cdrom/oracle805/orainst**, and enter one of the following commands to start the Oracle installer.

```
./orainst /m      # for the Motif-based graphical interface
./orainst /c      # for the character-terminal-based interface
```

5. If Oracle has not yet been installed, choose Default Install, then Install, Upgrade, or Deinstall Software, and then Install New Product with Database Objects. If Oracle has been installed, choose Add/Upgrade Software.
6. Confirm the pathnames of the ORACLE_BASE and ORACLE_HOME directories, and Oracle instance name, which is the value of the ORACLE_SID environment variable.
7. Confirm the location of the installation-log files.
8. Select the following products to install. Some options reside within categories in the Oracle Software Asset Manager dialog. The names of these categories, such as Oracle 8.0.5 Options, are prefixed with a plus sign (+). To expand a category and select from among the options it contains, double-click the name of that category. Then select the options of your choice.
 - Net8 Version 8.0.5.0.0
 - Net8 External Naming Adapters (entire category)
 - Net8 Protocol Adapters (entire category)
 - Oracle Advanced Networking Version 8.0.5.0.0 (entire category)
 - Oracle On Line Text Viewer
 - Oracle Partitioning Option (in the Oracle Options category)
 - Oracle UNIX Installer
 - Oracle8 Enterprise RDBMS 8.0.5.0.0
 - PL/SQL 8.0.5.0.0
 - Solaris Documentation
 - SQL*Plus

You can ignore any warning messages that appear regarding previously installed storage-manager products.

9. Choose **OK** when you are asked to confirm default database start-up.
10. When you are prompted to do so, enter the OFA-compliant mount points that you have defined for the datamart data, redo logs, and archive logs:

| | |
|------|-----------------------------------------|
| /u01 | # Oracle system tables |
| /u02 | # Oracle system rollback segments |
| /u03 | # Datamart tables |
| /u04 | # Additional datamart tables (optional) |

11. Enter a suitable pathname for Oracle documentation when prompted. E.piphany suggest entering a pathname of the form:

`/ORACLE_HOME/doc`

Replace `ORACLE_HOME` with the pathname of the home directory for your Oracle instance.

Note: At this point, the Oracle installer begins to download files. This process typically takes less than an hour, and requires periodic checking for status or error messages. The progress bar that the installer displays does not extend fully across the box even though all files have been downloaded. This is not an error. You must take the steps that follow to complete the Oracle installation procedure.

12. When the installer displays the Information dialog box, choose **OK** in it and all subsequent dialog boxes, then exit the installer.
13. If you are adding a new Oracle instance to a host on which Oracle already resides, back up the **listener.ora** file and the **tsnames.ora** file, then edit these files to include entries for your new instance.

INSTALLING THE ORACLE 8.0.5.1.0 (OR HIGHER) PATCH

Take the following steps to install the Oracle 8.0.5.1.0 (or higher) patch:

1. Log into the Download section of the Oracle MetaLink Web site, then enter the following information. (This is a password-protected site. Contact your Oracle sales representative to obtain access to this site.)
 - Product: Oracle RDBMS
 - Platform: Sun SPARC Solaris
 - Sort by: Last update date
2. Choose patch set 80510 (or higher) from the list of available packages.
3. Follow the instructions that are provided in the accompanying README file to download and install the patch.

RUNNING THE ROOT.SH SCRIPT

If you are installing an Oracle8 database server for the first time, follow the steps below to run the **root.sh** script. Otherwise, you can proceed to the next section.

*Warning: If you have upgraded from an earlier version of Oracle8, **stop**. Do not proceed with the following steps. Instead, continue with the next section, "Configuring Oracle," on page 38.*

1. Log in as **root** and enter one of following shell commands to clear environment variables that might adversely affect the installation process:

```
unset SRCHOME TMPDIR TWOTASK      # Bourne or Korn shell
```

```
unsetenv SRCHOME TMPDIR TWOTASK   # C shell
```

2. Review the **root.sh** script, and if it is correct, run it. If not, you can update this script and then run it without having to rerun the installer:

```
cd $ORACLE_HOME/orainst
sh ./root.sh
```

3. Answer Y to the following prompt if it appears:

```
ORACLE_HOME does not match the home directory for Oracle.  
OK to continue? [N]:
```

Note: At this point, you might see a warning to the effect that the ulimit has been exceeded. You can ignore this warning.

CONFIGURING ORACLE

E.piphany suggests that you take the following steps to configure Oracle8 to support an EpiCenter datamart. For detailed configuration instructions for Oracle, refer to the chapter entitled “Configuring the Oracle8 System” in the *Oracle8 Installation Guide*.

1. Log in as **oracle**.
2. Download the E.piphany configuration scripts for Oracle, as follows:
 - a) Insert the E.piphany installation disk in the CD-ROM drive on your Solaris host.
 - b) Enter the following commands:

```
cd $ORACLE_HOME/rdbms/admin  
cp /cdrom/resources/* .  
cd $ORACLE_BASE/admin/$ORACLE_SID/pfile  
cp $ORACLE_HOME/rdbms/admin/init$ORACLE_SID.ora .
```

3. Ensure that the Oracle instance for your EpiCenter datamart is running:

```
/bin/ps -ef | grep pmon
```

If the instance is running, the **ps** command displays a process listing that includes the SID for your Oracle instance.

If your instance is *not* running, take the following steps to start it:

- a) Enter the **svrmgrl** command:

```
svrmgrl # starts the server manager
```

b) Enter the following commands in **svrmgrl**:

```
connect internal
startup
disconnect
exit
```

4. Check to see that the Oracle listener process is running by entering the following command:

```
lsnrctl status
```

If this process is not running, **lsnrctl** displays a number of “unable to connect” and “no listener” messages, among others. To start the listener process, enter the following command:

```
lsnrctl start
```

5. Verify that there is an entry for your EpiCenter database in the **/var/opt/oracle/oratab** file of the following form. If you are adding an instance to a host on which Oracle already resides, add an entry for the new instance in addition to the entries for any previous instances that might already be present.

```
ORACLE_SID:ORACLE_HOME: Y
```

Replace **ORACLE_SID** with the value of the **ORACLE_SID** environment variable. Replace **ORACLE_HOME** with the value of the **ORACLE_HOME** environment variable.

Note: If the letter N appears at the end of the entry, replace it with Y to enable automatic start-up whenever the operating system reboots.

6. Make sure that the Oracle command file has correct permissions, as follows:

```
cd $ORACLE_HOME/bin
chmod 4751 oracle
ls -lg oracle
```

The correct permissions are:

```
-rwsr-x--x  oracle  dba  oracle
```

7. Enter the following commands to create symbolic links to the **\$ORACLE_HOME/dbs/init\$ORACLE_SID.ora** file for your Oracle instance.

```
cd $ORACLE_HOME/dbs
mv init$ORACLE_SID.ora init$ORACLE_SID.orig
ln -s ../rdbms/admin/init$ORACLE_SID.ora init$ORACLE_SID.ora
ln -s ../rdbms/admin/init$ORACLE_SID.ora \
    $ORACLE_BASE/admin/$ORACLE_SID/pfile/init$ORACLE_SID.ora
```

8. Edit your **init\$ORACLE_SID.ora** file to provide values that are consistent with the size of your application by following the instructions that are embedded in that file. Please refer to the *Oracle8 Reference* for detailed information about mount points, initialization parameters, and values.

Note: E.piphany applications typically perform large decision-support queries, as opposed to the large numbers of concurrent-but-brief queries of a typical on-line-transaction-processing (OLTP) application. E.piphany recommends that you configure Oracle using initialization parameters and values that are geared toward high TPC-D (Transaction Processing Performance Council benchmark D) performance. The sample file provides suggested values only, which E.piphany recommends that you edit to suit your application. You can adjust those parameters over time as you learn more about performance in your specific circumstances.

9. Enter the following commands to create symbolic links to the **\$ORACLE_HOME/rdbms/admin/config\$ORACLE_SID.ora** file:

```
ln -s ../rdbms/admin/config$ORACLE_SID.ora \
    config$ORACLE_SID.ora
ln -s ../rdbms/admin/config$ORACLE_SID.ora \
    $ORACLE_BASE/admin/$ORACLE_SID/pfile/config$ORACLE_SID.ora
```

10. If you are configuring an Oracle instance on a host on which Oracle already resides, enter the following commands to create administrative directories for your new instance:

```
mkdir $ORACLE_BASE/admin/$ORACLE_SID
mkdir $ORACLE_BASE/admin/$ORACLE_SID/bdump
mkdir $ORACLE_BASE/admin/$ORACLE_SID/cdump
mkdir $ORACLE_BASE/admin/$ORACLE_SID/pfile
mkdir $ORACLE_BASE/admin/$ORACLE_SID/vdump
```

CREATING CONTROL FILES

E.piphany provides a sample SQL script, called **epi_idb.sql**, that you must edit and then run to create to control files and perform other initialization tasks. Step 2, on page 38 describes how to download this and other sample scripts.

```
-- epi_idb: create system tables, control files, and rollback
segments
-- Replace all occurrences of 'SID' with the SID of your Oracle
instance.
-- Be sure to edit pathnames. Names are case sensitive.

connect internal;
set echo on;
set termout on;
spool dbcre.log;
shutdown;

-- initialization file
startup pfile=/u01/app/oracle/admin/SID/pfile/initSID.ora nomount
select to_char(sysdate, 'MM-DD-YYYY HH24:MM:SS') now from dual;
create database
    controlfile reuse

-- redo log files:
logfile '/u02/app/oradata/SID/redoSID01.log' size 100m reuse,
        '/u02/app/oradata/SID/redoSID02.log' size 100m reuse

-- system tablespace file:
datafile '/u01/oradata/SID/system01.dbf' size 180m reuse
maxdatafiles 100;
create public rollback segment RS1
storage(initial 300K next 300k);
```

```
alter rollback segment RS1 online;
shutdown;

-- startup pfile:
startup pfile=/u01/app/oracle/admin/SID/pfile/initSID.ora
select to_char(sysdate, 'MM-DD-YYYY HH24:MM:SS') now from dual;
spool off;

@catalog.sql;
@catparr.sql;
@catproc.sql;
@utlxplan.sql;
@catldr.sql;

exit;
!EOF

spool off;
```

To edit this script, you must replace every occurrence of *SID* with the SID of your Oracle instance. Please be aware that quoted strings and pathnames are case sensitive in Oracle SQL. To avoid case-sensitivity problems, E.piphany tablespace names and filenames are all uppercase.

When you have edited the script, you can run it using **svrmgrl**, as follows:

1. Log in as **oracle**, then enter the following shell commands:

```
cd $ORACLE_HOME/rdbms/admin
svrmgrl
```

2. When you see the **svrmgrl** prompt, enter the following commands:

```
connect internal;
@epi_idb.sql
exit
```

Note: The **epi_idb.sql** script calls several system-catalog-creation scripts that produce copious status messages and “drop object” warnings that you can ignore.

ASSIGNING DATA STORES IN ORACLE

EpiPhany provides a UNIX (Bourne) shell script, called **epi_tsp.sh**, that you can use as an aid in configuring tablespaces for your EpiCenter datamart. Based on your input, this shell script produces an SQL script that contains the appropriate CREATE TABLESPACE commands for your datamart. If you have not already done so, follow the instruction in Step Step 2, on page 38 of the Configuring Oracle section to download this script.

Use the **epi_tsp.sh** shell script to specify the number of files to create for the following tablespaces:

- Large fact tables
- Indexes on large tables
- Dimension tables
- Indexes on dimension tables
- Metadata tables
- Transient tables
- Other application-specific tables

Take the following steps to use this script:

1. Log in as **oracle**, then enter the following command:

```
$ORACLE_HOME/rdbms/admin/epi_tsp.sh
```

The **epi_tsp.sh** script issues a series of prompts to guide you through the process of creating tablespaces for your datamart. These prompts indicate default values for sizes and file counts that are calculated based on the initial settings that you specify for the file size and number of files in the fact-table tablespace. In some cases, such as for the metadata tablespace, file sizes are fixed.

2. Edit the **epi_tsp.sql** script to ensure that the sizes and pathnames for datafiles are correct. EpiPhany recommends that you avoid reducing the file size for the metadata tablespace below 100 megabytes.
3. Enter the following shell command:

```
sqlplus internal
```

4. Enter the following **sqlplus** command:

```
@epi_tsp.sql
```

Note: This SQL script creates tablespaces with specific names that EpiManager and EpiChannel recognize by default. If you use alternate names for tablespaces, you must define values for the macros that EpiChannel uses to locate those tablespaces. Refer to the EpiPhany System Guide for details on E.piphany macros.

CREATING EPIMART AND EPIMETA USERS

E.piphany provides a sample SQL script, called **epi_usr.sql**, that you can use to create standard users (owners) for EpiCenter datamart and metadata tables. You must edit this script to specify the passwords for the EPIMETA and EPIMART users. After you have done so, you can take the following steps to run this script:

1. Enter the following shell command:

```
sqlplus internal
```

2. Enter the following **sqlplus** command:

```
@epi_usr.sql  
exit
```

Warning: You can perform this and other Oracle configuration task with the Security Manager utility of the Oracle client package for Windows NT Server. If you do use the client package to administer your Oracle database, do not open the Enterprise manager as user EPIMETA. Enterprise Manager creates spurious tables in the EPIMETA tablespace.

This completes the configuration instructions for Oracle and the preparations required for the datamart host. For information on preparing your E.piphany application host, please proceed to the next section.

PREPARING YOUR APPLICATION HOST

E.piphany applications require a minimum of 64 megabytes of RAM on your application host. You must install the following software packages on your application host before you install E.piphany software:

- Windows NT Server 4.0
- Microsoft Internet Information Server (IIS), Version 4.0
- Selected components of Microsoft Office

If your application host is not the same computer as your datamart host you must also:

- Install the appropriate client software for the database server on which your datamart resides.
- Configure connectivity for your client software.

The following sections describe these activities.

INSTALLING WINDOWS NT SERVER 4.0 ON YOUR APPLICATION HOST

If you plan to install your E.piphany application on the same host as your datamart, you do not need to install Windows NT Server again. Otherwise, please see, “Windows NT Server 4.0 Installation,” on page 18, and “Network Connectivity,” on page 20 for instructions on installing and configuring this operating system.

INSTALLING INTERNET INFORMATION SERVER, VERSION 4.0

Epiphany requires that you install Microsoft Internet Information Server (IIS) 4.0. IIS 4.0 is distributed with Windows NT Server, Options Pack 4.0. Detailed instructions for installing components of this options pack are provided on the installation disk. You can install additional components of the options pack on your application host if you choose.

Follow these recommendations when setting up IIS:

- Use the default values for the Options dialog box.
- You may specify different values in the Database Publishing Directory dialog box. To use the security features of IIS, be sure that the **WWWroot** directory resides on an NTFS file system.
- Select the Microsoft SQL Server driver to install.
- Set the Time Zone.
- In the Windows NT Server Services dialog, set the following services to Manual start-up: Certificate Authority and Content Index.

After installing the package, please reboot Windows NT Server.

For further information about IIS 4.0, please refer to the Microsoft support site:

<http://www.microsoft.com/support>.

INSTALLING MICROSOFT OFFICE COMPONENTS

You can install Microsoft Office from the Microsoft Office CD. The installation program provides detailed instructions. Please follow those instructions to install the Microsoft Exchange mail client, ODBC connection client, and any other components of Microsoft Office that you choose to include.

INSTALLING AND CONFIGURING CONNECTIVITY PACKAGES FOR YOUR DATABASE CLIENT SOFTWARE

If you plan to install E.piphany software on the same computer as your datamart, you do not need to install or configure additional client software. You can instead turn to Chapter 2, “Installing E.piphany Software.” If you plan to install your E.piphany application on a separate computer, please proceed with the sections that follow.

SQL UTILITIES

If you are using SQL Server for your datamart, you must install the SQL Utilities, option on the SQL Server installation CD, which provides appropriate instructions.

ORACLE8 UTILITIES

If your datamart resides on an Oracle database server, you must install the following utilities:

- Oracle8 Client for Windows NT Server
- Version 8.0.4.4.0 of the Oracle ODBC driver

Please take the following steps to install these packages:

1. In the Select Installation Options dialog, choose Oracle8 Client.
2. In the Select Oracle8 Client Configuration dialog, choose Database Administrator.
3. Install the Oracle8 Client package:

The Oracle installer installs Net8 as part of the client-utilities package. You use Net8 to establish connectivity with your datamart. To establish a connection to the Oracle instance on which the datamart resides, you must provide Net8 with the following information about that instance.

You can use the Net8 Easy Config utility that comes with Oracle8 Client, or you can update the **tnsnames.ora** file directly. Either way, you must supply the following information:

- The protocol, in this case TCP/IP
- The hostname or IP address of the datamart host
- The port number of the TNS listener service on the datamart host
- The Oracle instance name (SID)

The **tnsnames.ora** file is located in the **Orant\Net80\Admin** folder in your Oracle installation directory. A typical entry for an E.piphany datamart takes the following form:

```
ORCL=
  (DESCRIPTION=
    (ADDRESS=
      (PROTOCOL = TCP)
      (HOST = hostname)
      (Port = port)
    )
    (CONNECT_DATA = (SID = ORCL)
  )
)
```

Replace *hostname* with the name of the computer on which your Oracle database server resides. Replace *port* with the port number for your database server. For Oracle, the default port number is 1521. Replace each occurrence of *ORCL* with the actual SID of the instance on which your EpiCenter datamart resides.

You must ensure that the Net8 entry on the application host matches the Net8 entry for your EpiCenter instance on the datamart host. If necessary, you can create a Net8 alias for the EpiCenter instance on the datamart host. For additional details on configuring Net8, refer to the *Net8 Getting Started* manual.

4. Use the Oracle installer to install Version 8.0.4.4.0 of the Oracle ODBC driver, which you can download from the following Web site:

http://www.oracle.com/products/free_software/index.html

This completes the tasks that you must perform to prepare your E.piphany application host. Please turn to Chapter 2 for instructions on installing E.piphany software.

INSTALLING E.PIPHANY SOFTWARE

This chapter describes the procedures for installing and configuring E.piphany application software.

Warning: If you are upgrading from an earlier version of E.piphany software, you must export EpiMeta metadata from your existing E.piphany application before you install the current version of E.piphany software. Refer to “Exporting Metadata,” on page 61 for details.

The standard E.piphany software installation includes:

- the EpiManager™ administrative utility for configuring and managing your datamart and applications.
- the AppServer™ application server, which coordinates user requests for data and reports, returns results in HTML format, and supports navigation between E.piphany Web pages.
- the EpiChannel™ extraction facility for downloading data from source systems into your EpiCenter datamart.
- other E.piphany components and utilities.
- drivers and other third-party software components on which E.piphany software depends.

Note: E.piphany application software does not have to be installed on the same computer on which your EpiCenter datamart resides.

The computer from which you run the E.piphany Application Server (AppServer™) must provide:

- Windows NT Server 4.0 set to run in 256-colors mode or higher
- 64 megabytes of RAM for use by E.piphany software

- Microsoft Internet Information Server (IIS), Version 4.0

If IIS is not already installed on your application host, you must install it before you attempt to install E.piphany software. Please review the guidelines for installing IIS listed in “Installing Internet Information Server, Version 4.0,” on page 45.

- Selected components of Microsoft Office

If the Microsoft Exchange Messaging client and the ODBC connection client are not yet installed on your application host, you must install it. Please review the guidelines for installing these components described in “Installing Microsoft Office Components,” on page 46.

If your EpiCenter datamart resides on a Windows NT Server host, you can use the same computer for your datamart and your E.piphany application.

The E.piphany installer offers to install several third-party system components that are required to support your E.piphany applications if they are not already present on your system. These components include:

- ODBC and JDBC drivers
- Sun JRE
- Microsoft Internet Explorer 4.0 (IE), including the Microsoft Java virtual machine

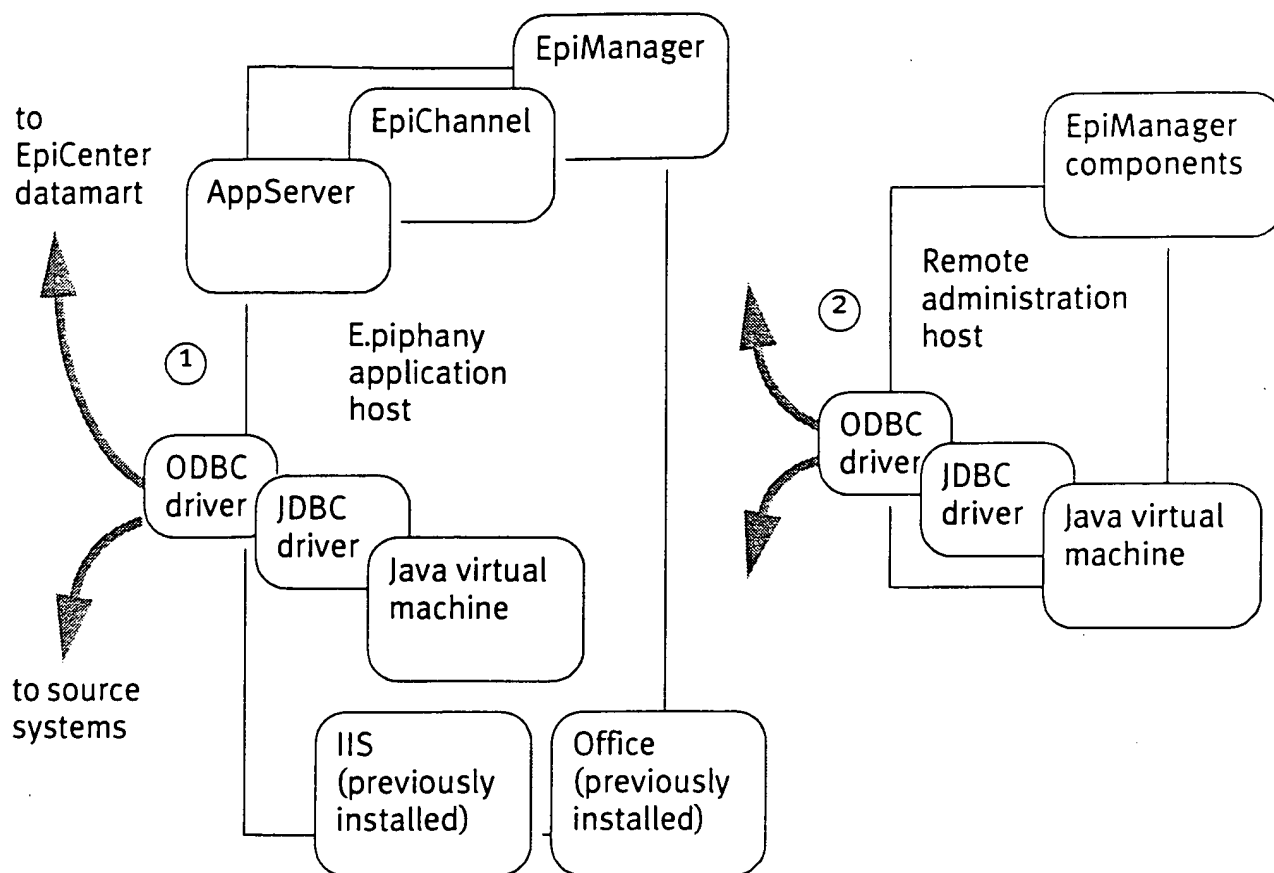
Please install IE 4.0 if the installer asks you to do so. The installer prompts you to exit if the Java virtual machine is not present and you do not install it.

After you have installed E.piphany software on the on the application host, you can install individual components of the EpiCenter Enterprise Manager (also referred to as EpiManager™) administrative utility on other computers to provide easier access for specific administrative tasks.

To install the complete set of standard E.piphany software on your application host, choose the Application Server installation option. To install an additional copy of selected E.piphany utilities on a remote host (which can run Windows 95, Windows 98, or Windows NT Server 4.0), choose the Remote Administration option.

Figure 2 illustrates the components that you install on the E.piphany application host and any remote-administration hosts that you might add.

FIGURE 2: EPIPHANY APPLICATION COMPONENTS



PERFORMING THE INSTALLATION

Take the following steps to install Epiphany software on your application host:

1. Log in to an account that has administrator privileges on the local host.
2. Exit all open Windows applications.

If an AppServer instance is running, from the Start menu, go to **Settings\Control Panel\Services** and manually stop the Application Server. If you do not, the installer cannot properly download the DLLs that this release requires.

3. Insert the Epiphany CD into your CD-ROM drive.
4. Double-click the **Setup.exe** icon in the CD-ROM folder.
5. If the Microsoft Java virtual machine is not installed on your system, you are requested to install it. Please choose Yes in response to this prompt.
6. The System Configuration screen displays your system's configuration data (for your information).

Follow the instructions below for either the Application Server or Remote Administration installation options. If you are installing E.piphany software for the first time, choose the Application Server option.

Warning: The installation program prompts you to reboot when the installation process is complete. If you answer Yes, you must again log in to a user account with administrator privileges to ensure that all new versions of .dll and other files are properly registered.

APPLICATION SERVER

The following steps apply to the Application Server installation option.

1. Choose the Application Server option.
A Services window shows which services will be stopped before the copy operation begins.
2. Enter the instance name. This is the name of the service as it will appear in the Service Control Panel. If you are upgrading from a previous version of E.piphany software, you must use a different instance name than that of your existing E.piphany instance.

An instance name distinguishes among multiple installations of the Epiphany software on the same machine. Typically, you will install the software once on a machine and thus have one instance. You are asked if this instance will be the default instance. If so, the **Setup** program makes this instance the default Web server destination.

3. Enter the machine name on which the Application Server runs.
By default, this is the full DNS name of the server. Because of your local network configuration, you may need to use a unique name; for example, the machine name without the domain name. Check with your local system network administrator if you have any questions.

4. Accept the default TCP/IP port for the Application Server if there is only one instance of the Application Server.

Each instance must have a separate TCP/IP port. Check with your local system network administrator if this is the case.

5. If your datamart resides on SQL Server, enter the name of the database server and the database name. The database name is the name of the EpiMeta database. This database does not need to exist as of yet. (The Registry keys are populated based on what you enter.)

If your datamart resides on an Oracle database, enter the hostname or IP address of the computer on which that database resides.

6. If your datamart is on SQL Server, enter the user name and password for the database server. The user must have system administrator (SA) privileges on the EpiCenter host.

If your datamart resides on Oracle, enter the user name for your EpiMeta user (typically **EPIMETA**) and the password for that user.

7. At this point, the installer asks if the instance name you entered in Step 1 is the default instance name. If you are installing E.piphany software for the first time, or if you plan to run a single instance of AppServer on this host, click Yes.

If you click No, your Web Server will not be set up to automatically route users to the Epiphany login page. In that case, users can access the login page by entering:

`http://machine_name/scripts/instance_name/Epiphany.dll`

8. Select the destination directory for the Epiphany software. The default directory is:

`C:\Program Files\Epiphany\instance_name`

9. Enter the location of the run-time reports and charts that end users will access. The default location is:

`C:\Program Files\Epiphany\instance_name\Charts`

10. Create a new folder as requested.

11. Select Administrative Tools from the components list. If you want to install Epiphany documentation on your hard disk, choose Documentation also.

- Administrative Tools

If you are installing E.piphany software for the first time, choose the components that have been selected by default.

- Documentation

Epiphany provides technical manuals on line in PDF format with keyword-search indexes. To use the PDF full-text index, you must have Acrobat Reader with Search or Acrobat Exchange. You can download Acrobat Reader 3.0 from the following Web site:

<http://www.adobe.com/prodindex/acrobat/readstep.html>

12. Select the default program-icon location.

13. If you are performing an upgrade from a previous E.piphany release and your datamart is already populated with data, please refer to Chapter 3, "Preparing to Migrate From Previous Releases," for further instructions.

This completes the Application Server installation option.

REMOTE ADMINISTRATION

1. Select the destination directory for the Epiphany software. The default is **C:\Program Files\Epiphany\instance_name**.
2. Enter the location of the run-time reports and charts that end users will access. **C:\Program Files\Epiphany\instance_name\Charts** is the default. Create a new folder as requested.
3. Select the Administrative Tools from the list of components to be installed: If you want to install Epiphany documentation on your hard disk, choose Documentation also.

- Administrative Tools

You have the option of selecting components of the following Administrative Tools: EpiCenter Manager, Web Builder, and Security Manager. Although EpiCenter Manager includes Web Builder and Security Manager, these components can be installed separately.

A person who does not need to have access to all of the features of EpiCenter Manager can use one of these components. The front-end configuration features of EpiCenter Manager are available through Web Builder. Access rights and system permissions can be administered using Security Manager.

- Documentation

Epiphany provides technical manuals on line in PDF format with keyword-search indexes. To use the PDF full-text index, you must have Version 3.0 or higher of Acrobat Exchange or Acrobat Reader. You can download Acrobat Reader 3.0 from the Adobe Web site:

4. Select the default program-icon location.
5. If you are performing an upgrade from a previous Epiphany release and your datamart is already populated with data, please refer to Chapter 3, "Preparing to Migrate From Previous Releases," for further instructions.

This completes the Remote Administration software-installation option.

CONFIGURING E.PIPHANY SOFTWARE

After you have installed E.piphany software, you must configure the E.piphany Web-server proxy. In addition to configuring the proxy, E.piphany also recommends that you enable performance monitoring for data-extraction operations.

CONFIGURING THE E.PIPHANY WEB-SERVER PROXY

The Epiphany Web-server proxy interacts with IIS 4.0. For the Epiphany Application Suite to function properly, set the IIS configuration values according to the instructions in this section.

1. Open the IIS 4.0 Service Manager from the Start menu by choosing Programs, then Windows NT Server 4.0 Option Pack, then Microsoft Internet Information Server, and then Internet Service Manager.
2. Right-click your server icon and select Properties from the pop-up menu. The Properties dialog box is displayed in the Master Properties panel. Select WWW Services as the Master Properties. Click Edit. In the WWW Service Master Properties dialog box, select Directory Security. In the Anonymous Access and Authentication Control Panel, click Edit. The Authentication Methods dialog box is displayed. Select Allow Anonymous Access. Click Edit.
3. Configure Anonymous Login with file access permissions for reading and writing to all Epiphany files. Enter the following attributes for the Anonymous user account:
 - Username
A username that has file access permission for all of the Epiphany installed files and directories. This username also needs permission to read the Epiphany entries in the Registry.
 - Password
The password for this username.

- Make sure that the directories that contain the **Epiphany.dll** and the **makechart.dll** files have execute permissions.

SETTING UP NT PERFORMANCE MONITORING FOR EXTRACTION

The **extract.exe** program is instrumented to use the standard NT Performance Monitoring facility, but you must take the following steps to enable the display of E.piphany data extraction processes in the NT Performance Monitor:

1. Your **Epiphany\instance_name\win32** installation directory must contain these items: **EpiPerfMon.dll**, **EpiPerfMon.ini**, and **EpiPerfMon.reg**.

EpiPerfMon.reg has values specified for the following Registry key:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\
instance\Performance
```

Enable write permission for the **EpiPerfMon.reg** file, then edit this file to ensure that the value of Library points to the correct folder in which **EpiPerfMon.dll** resides; that is, your current Win32 directory. Change the *instance_name* to the correct value for the Library path in this file.

2. Run **EpiPerfMon.reg**. To check the results, open the Registry and navigate to the key specified in Step 2. Ensure that the path supplied for Library is correct. If it is not, the Epiphany Channels object entry does not appear in the object list for the Performance Monitor.
3. From a command line, go to the **Win32** directory of the current instance and run the following commands to unload parameters from any previous Registry entry and initialize the values associated with your Epiphany application:

```
unlodctr Epi
lodctr EpiPerfmon.ini
```

Note: *Epi is the default driver and key specified in **EpiPerfmon.reg**.*

You can now start the NT Performance Monitor and monitor the progress of data-extraction jobs performed by EpiChannel.

VERIFYING YOUR INSTALLATION

When you install the Epiphany software, the first screen displays your system configuration.

After installing the Epiphany software you can:

- Run the Epiphany tools, such as EpiManager, to configure your EpiCenter and set up EpiChannel extraction jobs. You will also use EpiCenter to construct the topics and Web pages that enable users to query the data in your data warehouse.

The *Epiphany System Guide* gives instruction on how to use EpiManager to administer your E.piphany application. Please read Chapters 1 and 2 of that guide for background information before using EpiManager.

- Enter your e-mail password to receive notification of Epiphany system status. You will use the Configuration dialog box in EpiManager to set this up. Instructions for doing so appear in Chapter 3 of that guide.
- Run the E.piphany extraction program (EpiChannel) to extract data from your source systems and place them in your EpiCenter datamart. EpiChannel performs the extraction jobs that you defined in EpiCenter Manager.
- Start AppServer, the Epiphany Application Server. Refer to the *Epiphany System Guide* for details.

PREPARING TO MIGRATE FROM PREVIOUS RELEASES

This chapter describes the preparations you must make before you can upgrade from a previous E.piphany release to Release 4.0. These preparations include:

- Migrating an EpiMart database that resides on SQL Server to SQL Server Version 7.0
- Migrating EpiMart database that reside on Oracle to Oracle8 Version 8.0.5.1.0
- Exporting EpiMeta metadata from your earlier E.piphany release for later import into your Release 4.0 EpiMeta database
- Modifying extraction jobs to eliminate obsolete data types

This chapter describes each of these actions. Chapter 3 of the *E.piphany System Guide* describes the steps that you take to complete the upgrade process after you have installed E.piphany e.4 Release 4.0.

MIGRATING AN EPIMART DATABASE

The sections that follow describe the preparations you must take to migrate your datamart to a database server that E.piphany e.4 Release 4.0 supports.

SQL SERVER

If you are upgrading your EpiPhany application from Release 3.3 or earlier, and you plan to continue using SQL Server as the database server for your datamart, you must:

- Install and configure Version 7.0 of SQL Server if you have not already done so. Refer to Chapter 1, “Preparing to Install E.piphany Software.” for details.
- Truncate your existing staging tables to eliminate the possibility of null-value errors that could halt the process of upgrading your datamart. These staging tables contain temporary data only. Truncating them has no lasting effect on your datamart. Take the following steps:
 1. Using the **isql** utility, connect as user **dba** to the database that stores your EpiMeta metadata, then enter the following SQL commands:

```
select 'truncate table ' + stage_name
      from dim_base_view
      where dim_base_type = 'Default'

select 'truncate table ' + stage_name
      from fact_tbl_view

select 'truncate table ' + external_tbl_name
      from external_tbl
```

2. Save the output of the commands you ran in Step 1. This output takes the form of a set of SQL statements.
 3. Connect to the database that stores your EpiMart datamart data, then enter the output that you saved in Step 2 as commands in **isql**. These commands truncate the staging tables when run against the datamart.
- Follow the procedure provided by the SQL Server 7.0 installer to upgrade your EpiMart database.

ORACLE

If you are upgrading your E.piphany application from Release 3.4 or earlier, and you plan to continue using Oracle as the database server for your datamart, you must:

- Install and configure Version 8.0.5.1 if you have not already done so. Refer to Chapter 1, “Preparing to Install E.piphany Software,” for details.
- Follow the procedures described in the Oracle8 Installation Guide to upgrade your EpiMart database.

EXPORTING METADATA

This section describes the steps that you must take to preserve existing metadata prior to installing the current version of E.piphany software.

EpiManager provides a facility for upgrading your existing installation. Rather than operating on the EpiMeta database itself, the upgrade procedure uses a standard E.piphany export file in Microsoft Access format. You use this facility to export existing metadata from your older version of E.piphany software *before* you install your new version of E.piphany e.4 software. After you have installed the current version of E.piphany software, you can import the metadata from the export file and perform additional steps to optimize it for use with in current release

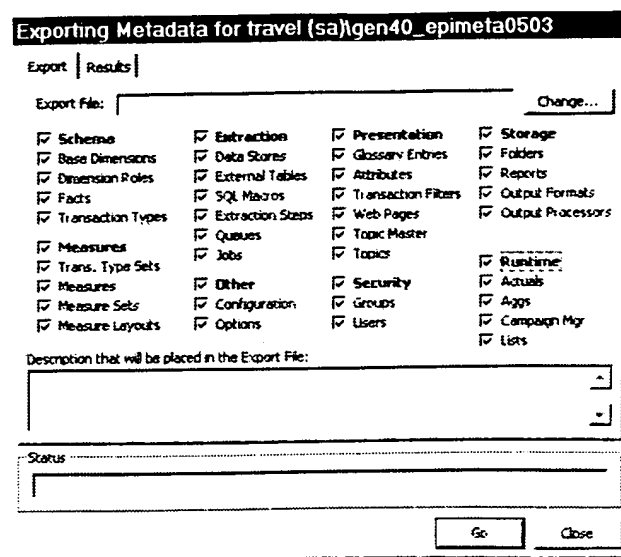
Take the following steps to export your existing metadata:

1. Use the **EpiCenter/Export/Export All** command in the EpiCenter Manager menu to generate a full export file of your existing metadata.

By default, the Actuals and Aggs are not exported when you issue the Export All command. If you intend to preserve an existing datamart, select *both* of these options in the run-time section of the Exporting Metadata dialog. (See Figure 3, on page 62.) If you plan to create a new datamart, you do not need include these two options.

2. Use the **EpiManager\EpiCenter\Initialize EpiCenter** command in EpiManager to build a new EpiMeta database.

FIGURE 3: EXPORTING METADATA DIALOG BOX



Refer to Chapter 3 of the *Epiphany System Guide* for a complete description of the procedure that you follow to migrate the metadata and saved reports that you have preserved in the export file.

MODIFYING EXTRACTION JOBS

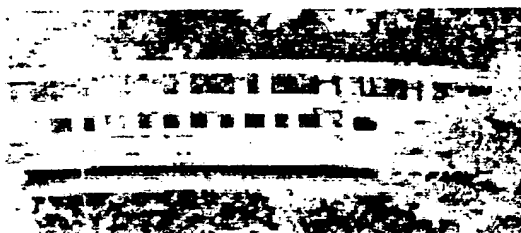
This release of E.piphany software no longer supports the following data types:

NUMBER(9)
NUMBER(9,2)
FLOAT

You must replace all references to these data types in extraction jobs with references to one of the following new physical data types before you can extract data into an E.piphany e.4 Release 4.0 datamart:

EPIINT
FACTMONEY
FACTQTY

0-0-E-TP-EIG-4.0



E:PIPHANY

www.epiphany.com

SEMANTIC TYPES

This appendix describes the dimension and fact semantic types.

DIMENSION SEMANTIC TYPES

The dimension semantic types are Slowly Changing Dimensions, Latest Dimension Value, First Dimension Value, Initial Load Dimension, and Rewrite Dimension.

SLOWLY CHANGING DIMENSIONS

A Slowly Changing Dimension is a dimension in which the attributes or hierarchy of the dimension can change over time, but historical data is not restated. Two examples follow:

- A national sporting goods chain has stores divided into three regions. On January 1, 1998, the chain reorganizes its regions, moving Denver from the Central to the Western region. Their 1998 sales forecasts take into account that the Denver store is in the Western region, but they do not want to recalculate forecasts and actuals from previous years. Using the Slowly Changing Dimensions semantic type, sales for Denver can be aggregated up to the Central region through 1997. Beginning January 1, 1998, Denver sales are applied to the Western Region.

- On May 7, 1999, one of this store's customers changes careers and doubles her income. Since the store analyzes customer purchases based on demographics, all of this customer's new purchases should be analyzed with her new demographic data. However, all of her previous purchases should still be analyzed with her old demographic data. Using the slowly changing dimensions semantic type, all of this customer's purchases are recorded based on her demographics at the time of the purchase.

The Slowly Changing Dimensions semantic type accomplishes the following logic:

- Rows with the same **sskey** in the staging table will be eliminated following in a "last in wins" rule. The last row is determined by the special column called **ikey**, which is created by EpiChannel and is automatically incremented during normal extractions. The highest **ikey** row for a given **sskey** will be accepted.
- New rows are created by searching through the dimension-staging table for new **sskey** values, or **sskey** values that have one or more dimension column changes from the last known values. Each of these cases creates a new row in the dimension table. The mapping row for that **sskey** points to the latest dimension row with that **sskey** value.
- In the EpiChannel log file, new **sskey** rows are reported in the **Inserted** column. The number of changed rows is reported in the **Modified** column. The number of rows in the staging table is reported in the **Processed** column.
- The key column of the dimension table becomes the primary-key index. Each dimension column is also Indexed (non-uniquely). The mapping table is indexed (primary key) on **iss**, **sskey**. You can use the Dimension Column dialog box in EpiCenter Manager to disable the indexing of individual columns.

iss is used when two source systems have the same **sskey** values (such as when the SAP and Vantive systems both have a Customer #2 record). **iss** is determined by the source-system identifier that is selected in the General tab of the Data Store dialog box.

- The column *dimension_key_REAL* (where *dimension* is the dimension column name) is set to the first key value for each **sskey**. In other words, when a new **sskey** is discovered, the **REAL** key is set to the new dimension key value. Subsequent dimension rows for this **sskey** will retain the original **REAL** key value.

Note the following considerations:

- Do not allow dimension column values to *oscillate* unpredictably. (See “First Dimension Value,” on page 451 for more information.) In particular, do not rely on **key** filtering of duplicate **sskey** values if two or more rows during a single extraction might have different values for one or more dimension columns. The reason is that a new row will be created in the dimension table for every extraction for which a change is recorded. Consequently, two values can “compete” with each other, forcing an unending sequence of row creation in the dimension table.
- Rows can be removed from dimension tables after they have been extracted with an explicit delete or truncation, or through use of the Initial Load Dimension semantic.
- The **UNKNOWN** dimension row always has a key value of 1. Since the **key** value is constant, the **UNKNOWN** row does not have an entry in the mapping table.
- Slowly Changing Dimension cannot be used in Initial jobs.

LATEST DIMENSION VALUE

The Latest Dimension Value updates rows instead of performing Slowly Changing Dimensions. It applies changes retroactively to a dimension. Thus, the changes take effect for all historical data, as well as for current and future loads.

For example, assume that a sporting goods store has a category historically called **Rollerblades**. Now that they are selling other brands, the store wants to change the category to **In-line Skates**. By using the Latest Dimension Value semantic type, this change can affect all of the historical data because all previous sales of **Rollerblades** are now labeled **In-line Skates**. Consequently, the store can compare year-to-year sales of all in-line skates.

This semantic type has the same duplicate **sskey** filtering as Slowly Changing Dimensions. Use this semantic type for an implementation that “restates history” when a source dimension table changes.

Note the following considerations:

- New **sskeys** are inserted in both the dimension table and mapping table. Existing **sskeys** with one or more changed dimension columns are updated in place in the dimension table (that is, the same dimension row is used) with the latest values.
- In the EpiChannel log file, new **sskey** rows are reported in the **Inserted** column. The number of changed rows is reported in the **Modified** column. The number of rows in the staging table is reported in the **Processed** column.
- Latest Dimension Value has the same indexing as Slowly Changing Dimensions.
- The **REAL** column is always equal to the dimension key.
- Latest Dimension Value has the same UNKNOWN mapping behavior as Slowly Changing Dimensions.
- Latest Dimension Value cannot be used in Incremental or Initial jobs.
- All fact aggregate tables must be rebuilt after the Latest Dimension Value semantic is used.

FIRST DIMENSION VALUE

The First Dimension Value semantic type ignores any changes to a dimension. Values present when the row was first inserted are preserved forever, regardless of any future changes. This semantic never modifies a row after it has been seen.

You might want to use this type if your source data comes from two systems that are not in complete agreement with each other. For instance, one system might have Customer #12345 as David Anderson, and another system might have the same customer as David Andersen. Ideally, you would determine which one was in error and correct it.

In the meantime, you could choose to apply the first value read and to ignore the other. (This is a good method for avoiding the *oscillation* problem mentioned on page 449.) If you were to use the Slowly Changing Dimensions semantic type in this case, there would be a race between the two source systems for each extraction, and (in the worst case), your dimension values could alternate between the two values with every extraction.

Note the following considerations:

- First Dimension Value has the same duplicate **sskey** filtering as Slowly Changing Dimensions.
- New **sskeys** are inserted in both the dimension table and the mapping table. Existing **sskeys** are ignored.
- In the EpiChannel log file, new **sskey** rows are reported in the **Inserted** column. The number of rows in the staging table is reported in the **Processed** column.
- First Dimension Value has the same indexing as Slowly Changing Dimensions.
- The **REAL** column is always equal to the dimension key.
- First Dimension Value has the same UNKNOWN mapping behavior as Slowly Changing Dimensions.
- First Dimension Value cannot be used in Initial jobs.

INITIAL LOAD DIMENSION

The Initial Load Dimension semantic type ignores the current datamart entries. (It is also the fastest semantic type).

Initial Load Dimension loads dimension without regard to any previously existing rows. This semantic type can be used for the initial load of the empty EpiMart, and also to completely reload a dimension, ignoring existing values. Using any other semantic type would require emptying the existing dimension table before beginning the extraction.

Note the following considerations:

- Initial Load Dimension has the same duplicate **sskey** filtering as Slowly Changing Dimensions.
- The existing dimension and mapping tables are ignored; all **sskeys** are imported directly into the dimension and the mapping tables.
- In the EpiChannel log file, new **sskey** rows are reported in the **Inserted** column. The number of rows in the staging table is reported in the **Processed** column.
- Initial Load Dimension has the same indexing as Slowly Changing Dimensions.
- The **REAL** column is always equal to the dimension key.
- Initial Load Dimension has the same **UNKNOWN** mapping behavior as Slowly Changing Dimensions.
- Initial Load Dimension cannot be used in Incremental or Merge jobs.
- Running Initial Load Dimension on a dimension that has the **indiv** or **group** dimension role invalidates all saved lists that were generated from that dimension.

REWRITE DIMENSION

The Rewrite Dimension semantic type can be used to make a slowly changing dimension table act like a latest-dimension-value table. Rewrite Dimension first merges values in the staging table into datamart tables using the same logic as the slowly changing dimension semantic. For every **sskey**, it then finds the most recent row with that **sskey** and rewrites every other row with that **sskey** using the dimension column values of the most recent row. No dimension table rows are deleted, so fact tables are still valid. However, the meaning of fact rows may have changed. Since the meaning of fact rows may have changed, a full aggregate build is required on the history tables after running this semantic.

Note that the Rewrite Dimension semantic type can only be used in Rebuild and Non-partitioned jobs.

BACKFEED DIMENSION

The Backfeed Dimension semantic forms a union of the data in the active backfeed export table (P or Q) for a dimension and the data in the main active backfeed table (A or B) for that dimension. The resulting table is then placed in the inactive backfeed table, and the inactive backfeed export table is truncated.

FACT SEMANTIC TYPES

The fact semantic types are Transactional, Transactional/State-like, Transactional/State-like/Force Close, Pipelined, Initial Load Fact, First/Last Fact, Reload Date Fact, and Count Unjoined.

Note: Fact rows with all zero facts are discarded by all semantics except Initial Load Fact.

TRANSACTIONAL

The Transactional semantic type is the simplest means of moving fact-staging data into fact base tables. This semantic type uses the following logic:

- Only rows with **process_key** set to 1 (Transactional) are used; others are discarded.
- For **sskeys** that are already entered in the current-fact base table, all rows in the staging table with that **sskey** and with the same or earlier **date_key** are discarded. If the **sskey** already exists in the fact table, only later dates can be added to the fact table.

Two or more rows with the same **sskey** can be imported into the fact base table if they arrive in the same extraction and the **sskey** either does not already exist, or exists but is dated earlier. (This property is used by the *pseudo-order* approach to the Booking/Shipping problem; see “Transactional/State-like/Force Close,” on page 456.)

- In the EpiChannel log file, all rows added to the fact base table are reported in the **Inserted** column, and the total number of rows in the staging table is reported in the **Processed** column.

Note the following considerations:

- Transactional semantics should be used for event facts; once the fact event has occurred, it can never be modified.
- To reload transactions after they have been loaded into the fact base table, the rows must be deleted from the fact base table, or the fact base table must be truncated. See *Reload Date Fact* and *Initial Load Fact* for more information.
- The date comparison is always done with respect to **date_key**, which references the built-in date dimension role. Any user-defined dimension roles that refer to the date dimension are ignored.
- The Transactional semantic type cannot be used in an Initial job.

TRANSACTIONAL/STATE-LIKE

The Transactional/State-like semantic type allows changes to already existing rows in EpiMart. It uses the same logic as the Transactional semantic type, with the addition of the logic described below.

First these three steps occur:

1. Records in the staging table with **process_key** of 2 (state-like) are treated as Orders that can be *differenced* between extractions (a discussion of *differencing* follows).
2. For records in which **process_key** = 2, duplicate **sskeys** with the same **date_key** in the same extraction cause only the highest **ikey** value to be used. Other rows are discarded. This is the same type of filtering that occurs in dimension semantics such as Slowly Changing Dimensions.
3. For an **sskey** with the highest **ikey** (for every set of rows with the same **sskey**, the row with the highest **ikey** takes precedence), if the **date_key** for that staging row is less than the last **date_key** for that **sskey** in the fact base table, the staging row is discarded. In other words, an Order can only be modified on its last reported date, or some time further into the future.

After Steps 1-3, the staging fact columns and dimension values are compared with the current values in the fact table (if any). Adjusting records are created in the fact table so that the fact table now reflects the reported state from the staging table. (This is why the term *state-like* is used.) *Differenced* transactions are invented if the numeric fact columns have changed.

If the dimensionality has changed, then the Order is “debooked” and then “rebooked” with the correct dimensionality.

If the same **sskey** appears in the staging table with more than one **date_key**, then further adjusting transactions are made in the fact base table as appropriate to bring that table in line with the reported staging rows.

Note: By convention, Bookings are entered with positive facts (negative for Returned Orders), whereas Shipments are entered with negative facts (positive for Returned shipments).

When using this semantic type it is difficult to ensure that Backlog calculations will remain consistent when Orders are not completely closed in the source system. Use the Transactional/State-like/Force Close semantic type instead.

Note that the Transactional/State-Like semantic type cannot be used in an Initial job.

TRANSACTIONAL/STATE-LIKE/FORCE CLOSE

This semantic type is equivalent to Transactional/State-like with the following additional logic:

Once a Booked Order is entered into EpiMart, it remains in that state (with an Open Backlog) forever. In normal scenarios, an invoice eventually arrives, which will close the Backlog. However, in some source systems, Booked Orders can be removed from the system completely. If such an Order had been entered previously into EpiMart, then it will remain in an Open Backlog condition forever.

The solution to this problem is to use Transactional/State-like/Force Close, which establishes a “Challenge Protocol” for Open Orders. In this scenario, all Open Bookings must be extracted into the staging table during every extraction. The reason is that the Force Close logic will close out all Open Orders (**sskeys** with non-zero facts in the system) that do not appear in the fact-staging table. Only Booking Transaction types (those with **transtype_key** values between 1 and 99) will be affected in this manner.

To determine whether an order with a given **sskey** is open, the fact values of all rows with that **sskey** are added. This sum is taken over all values of **transtype_key**. If the sum is not zero, then the order is open. If bookings are recorded with positive fact values and shipments are recorded with negative fact values, then the open orders will be exactly those orders for which bookings are not equal to shipments.

When an order is closed by this semantic type, an order entry is constructed with the lowest **transtype_key** value that has been used for this **sskey**. Typically, this **transtype_key** is used for bookings. The fact values in this constructed order entry are those values that result in a sum of zero for the fact table rows with the **sskey** of the order being closed.

When using this semantic type, the methodology that has been found to work in practice involves the use of pseudo-orders. In this methodology, two **sskeys** are used for each order: one **sskey** for the original booking and a second **sskey** for both the shipments and the pseudo-orders. After an order is booked, the following extraction steps are used in every extraction, as long as the order is open:

1. One extraction SQL statement extracts all Open Orders. Fact amounts are the Open amounts (what has not been shipped yet), not the ordered amounts. The transaction type for this statement should be in the Booking range (1...99), the **process_key** value should be 2 (state-like), and the **sskey** value should be the same as that of the original order.
2. The second extraction statement extracts all new shipments. These shipments are recorded with a **process_key** of 1 (Transactional) and with the second **sskey** value. The fact values of these records should be negative for positive shipments. The transaction type should be in the shipment range (101 or greater).
3. The third statement is a restatement of the shipment, but as a Booking (**transtype_key** between 1...99). The **process_key** is still 1 (allowing the Transactional Semantics to import it), but the transaction type is a Booking. The same **sskey** and dimensionality as in the second statement are used. These are the pseudo-orders, since they are actual shipments that are entered as Orders.

The net effect of this methodology is as follows: When a shipment occurs, the open booking quantity (which is extracted in statement 1) is reduced by the number of units shipped. The state-like semantics then creates a difference transaction that reduces the amount of the open order to the new level. At the same time, a pseudo-order (with a new **sskey**) is created for the shipped amount by statement 2, and a (negative) shipment in the same amount is created by statement 3. Note that the convention of recording shipments as negative values ensures that the sum of fact values for the second **sskey** is always zero. Eventually, when the order is removed from the system, no state-like fact for that **sskey** is generated in statement 1. At this point, if the sum of values for the **sskey** is not zero, then a force-close transaction with values that result in a sum of zero is added to the fact table. Now the fact table has a value of zero for the original **sskey**, and booked and shipped values for the new **sskey** that correspond to the amount actually shipped.

For example, Table 28 illustrates the record for an order that might appear in a staging table.

TABLE 28: ORDER RECORD IN STAGING TABLE

| sskey | Cust | Prod | Date | trans | proc | Qty |
|-------|------|------|---------|-------|------|-----|
| 123a | 23 | 44 | 4/11/99 | 1 | 1 | 22 |

Table 29 illustrates how this data is merged into the fact table with transactional logic.

TABLE 29: ORDER RECORD IN FACT TABLE

| sskey | Cust | Prod | Date | trans | Qty |
|-------|------|------|---------|-------|-----|
| 123a | 23 | 44 | 4/11/99 | 1 | 22 |

Table 30 shows the result of extracting the records for 10 items that have been shipped to the customer:

TABLE 30: EXTRACTED FACTS FOR SHIPMENT

| sskey | Cust | Prod | Date | trans | proc | Qty |
|-------|------|------|---------|-------|------|-----|
| 123a | 23 | 44 | 4/14/99 | 1 | 2 | 12 |
| 234b | 23 | 44 | 4/14/99 | 101 | 1 | -10 |
| 234b | 23 | 44 | 4/14/99 | 1 | 1 | 10 |

The first row is a state-like fact for the part of the order that has not yet been shipped. The next two rows are the shipping fact and pseudo-order booking for the part of the order that was shipped. These facts are then merged into the fact table with transactional and state-like logic, as Table 31, on page 459 illustrates.

TABLE 31: SHIPMENT MERGED INTO FACT TABLE

| sskey | Cust | Prod | Date | trans | Qty |
|-------|------|------|---------|-------|-----|
| 123a | 23 | 44 | 4/11/99 | 1 | 22 |
| 123a | 23 | 44 | 4/14/99 | 1 | -10 |
| 234b | 23 | 44 | 4/14/99 | 101 | -10 |
| 234b | 23 | 44 | 4/14/99 | 1 | 10 |

If, in the next extraction, there is no record for **sskey** number 123a, then the force-close logic constructs a record to close the order (Table 32).

TABLE 32: FACT TABLE WITH FORCED-CLOSED ORDER

| sskey | Cust | Prod | Date | trans | Qty |
|--------------|-------------|-------------|----------------|--------------|------------|
| 123a | 23 | 44 | 4/11/99 | 1 | 22 |
| 123a | 23 | 44 | 4/14/99 | 1 | -10 |
| 234b | 23 | 44 | 4/14/99 | 101 | -10 |
| 234b | 23 | 44 | 4/14/99 | 1 | 10 |
| 123a | 23 | 44 | 4/17/99 | 1 | -12 |

The sum of the **Qty** field for all facts with **sskey** 123a is now zero, and the booking and shipping records are in the fact table with **sskey** 234b.

Note that the Transactional/State-Like/Force Close semantic type cannot be used in an Initial job.

PIPELINED

Use the Pipelined semantic type for facts that can exist in several different life-cycle phases, called pipeline states; for example, sales opportunities or support calls facts.

The **transtype_key** field represents the pipeline. Before using this semantic type, you must define the pipeline stages and numbers.

Pipelined generates these transactions:

1. When an **sskey** first enters some pipeline stage (**transtype_key**), a positive fact row is created with that **transtype_key**.
2. If that **sskey** subsequently moves backwards in the pipeline (that is, if the same **sskey** appears with a smaller transtype value), then in addition to the new row created by the previous step for the new pipeline stage, a negative “de-booking” transaction is created with the old **transtype_key** minus 1. (This is a loss transaction.)
3. If that **sskey** subsequently moves forwards in the pipeline (that is, if the same **sskey** appears with a larger transtype value), then in addition to the new row created in Step 1 for the new pipeline stage, a negative “de-booking” transaction is created with the old **transtype_key** plus 1. (This is a shipment transaction.)

Note: The transtype values for all pipeline stages must be multiples of 3. The +1 and -1 transtype values used in steps 2 and 3 above are created automatically from these values.

For example, assume that the pipeline consists of the following steps: Prospect, Lead, Quote, and Order. You could set up **transtype_key** fields for this pipeline as follows: 303= Prospect, 306 = Lead, 309 = Quote, and 312 = Order. The semantic type creates its own transactions for 302, 304, 305, 307, 308, 310, 311, and 313, which correspond to the forward and backward movements from the various pipeline stages.

To define measures that extract information such as Number of New Leads, you must define transaction types with these new names and keys (using EpiCenter Manager’s Configuration dialog box; see Appendix B, “EpiCenter Configuration.”.)

*Note: The pipelined semantic types ignores the **process_key** field.*

Note that the Pipelined semantic type cannot be used in an Initial job.

INITIAL LOAD FACT

Similar to the Initial Load Dimension, the Initial Load Fact semantic type is the fastest way to load a fact table. It also has the special property that the active current and history tables are ignored so that this semantic type can reload an already-populated fact table. All existing rows, however, are lost. For this reason, Initial Load Fact is usually used during development when an installation is verifying whether a first extraction yields correct data. All data is loaded into the history table (X or Y), and the current table (A or B) is left empty.

Important: For first time extractions, you can use this semantic template in place of all other fact semantic types that load data when each **sskey** is being loaded into the staging table *only once*. This is because upon first extract, when an **sskey** is loaded only once, the special logic of the other semantic types (such as delta inference during Transactional/State-like) does not apply. If the first extraction does require multiple records per **sskey**, such as loading inventory values using Transactional/Statelike, then Initial Fact Load cannot be used.

Note that the Initial Load Fact semantic type cannot be used in Incremental jobs.

INITIAL LOAD FACT A/B

The Initial Load Fact A/B semantic type is the same as the Initial Load Fact semantic type, except that all data is loaded into the A or B table. This semantic type can only be used with the Non-Partitioned job type, and it is intended to be used in datamarts that do not make use of the history (X and Y) tables.

RELOAD DATE FACT

The Reload Date Fact semantic type is a hybrid of the Transactional and Initial Load Fact semantic types. Use it to reload a subset of an existing table in which the reload point is separable by date only.

Reload Date Fact first determines the minimum date that occurs in the fact-staging table. It then copies from the current fact table to the new fact table all existing EpiMart data that occurs with a date key prior to that minimum date.

The fact-staging table is then used to hold all subsequent dates. Thus, a complete load of data must be placed in the fact-staging data from the minimum date forward. When loading the fact-staging table (with SQL statements), use a WHERE clause based on a particular date.

Note that the Reload Date Fact semantic type cannot be used in incremental jobs.

RELOAD DATE FACT A/B

The Reload Date Fact A/B semantic type is the same as the Reload Date Fact semantic type, except that all data is loaded into the A or B table. This semantic type can only be used with the Non-Partitioned job type, and it is intended to be used in datamarts that do not make use of the history (X and Y) tables.

FIRST/LAST FACT

The First/Last Fact semantic type keeps track of the first and last fact values for an **sskey**.

For example, a store may wish to keep track of only the first and last purchases that a customer has made. This can be done with the First/Last Fact semantic type by using the customer ID as the **sskey**.

The First/Last Fact semantic type does not use the transaction type values in the staging table. Instead, it uses the **FIRST** and **LAST** transaction types. By default, the values for **FIRST** and **LAST** are 120 and 121, respectively. performs the following actions:

1. As with all other fact semantics, if two or more staging table rows have the same **sskey** and date values, only the last of these rows is used.
2. For every **sskey** in the staging table, the staging table row with that **sskey** and the earliest date value is found.
 - If the fact table does not contain a row with this **sskey**, then the staging table row is added to the fact table with a **transtype_key** value of **FIRST**.
 - If the fact table contains a row with this **sskey**, a **transtype_key** value of **FIRST**, and a later date, then the values in the fact table row are replaced with the values in the staging table row. The value of **transtype_key** is set to **FIRST**.
 - If the fact table contains a row with this **sskey** and an equal or earlier date, then the staging table row is ignored.
3. For every **sskey** in the staging table, the staging table row with that **sskey** and the latest date value is found.
 - If the fact table does not contain a row with this **sskey**, then the staging table row is added to the fact table with a **transtype_key** value of **LAST**.
 - If the fact table contains a row with this **sskey**, a **transtype_key** value of **LAST**, and an earlier date, then the values in the fact table row are replaced with the values in the staging table row. The value of **transtype_key** is set to **LAST**.
 - If the fact table contains a row with this **sskey** and an equal or greater date, then the staging table row is ignored.

Note the following considerations:

- The First/Last Fact semantic type ignores the **transtype_key** and **process_key** values in the staging table.
- The First/Last Fact semantic type always adds two fact table rows for every new **sskey** in the staging table. If the new **sskey** only appears in one staging table row, then these two fact table rows are identical except for the value of **transtype_key**.
- The First/Last Fact semantic type cannot be used in Incremental or Non-Partitioned jobs.

COUNT UNJOINED

Count Unjoined informs you of the number of rows that are transformed by an outer join when facts are loaded from a staging table to the main fact tables. In order to prevent the loss of rows, outer joins are used to map fact-staging tables to dimension tables. Any unmatched dimension keys in fact-staging tables are automatically set to refer to the special UNKNOWN dimension value. The Count Unjoined semantic counts the number of rows that are transformed in this way and displays the result in the console window.

Note that the Count Unjoined semantic does not change any of the datamart tables.

BACKFEED FACT

The Backfeed Fact semantic forms a union of the data in the active backfeed export table (P or Q) for a fact and the data in the main active backfeed table (A or B) for that fact. The resulting table is then placed in the inactive backfeed table, and the inactive backfeed export table is truncated.

COMPARISON OF SEMANTIC TYPES

This section illustrates the differences between semantic types by means of examples from a fictitious datamart containing customer, product, and order data.

DIMENSION SEMANTICS

Table 33 and Table 34 show a set of rows that might appear in a pair of dimension staging tables after an initial extraction.

TABLE 33: INITIAL CUSTOMER STAGING TABLE

| key | sskey | iss | Customer_Name | Age | Date |
|------------|--------------|------------|----------------------|------------|-------------|
| 1 | bobb | 2 | Bob Bobster | 47 | 4/23/99 |
| 2 | lobt | 2 | Lobby Tableman | 15 | 4/7/99 |
| 3 | betk | 2 | Betty Kezer | 31 | 4/15/99 |

TABLE 34: INITIAL PRODUCT STAGING TABLE

| key | sskey | iss | Product_Name | Date |
|------------|--------------|------------|---------------------|-------------|
| 1 | cool | 2 | Cool Stuff 2 | 4/30/99 |
| 2 | wow | 2 | Wowzer Pro | 4/30/99 |
| 3 | neat | 2 | Neato Lite | 4/30/99 |

The **iss** key is a unique identifier for the source system type (the source system type is selected in the data store dialog box). The staging table rows are numbered sequentially by the value in the **key** column.

Table 35 and Table 36 show how these tables are loaded into the datamart with an Initial Load Dimension semantic.

TABLE 35: INITIAL CUSTOMER DIMENSION TABLE

| key | Customer_Name | Age | Date | key_REAL |
|-----|----------------|-----|---------|----------|
| 2 | Bob Bobster | 47 | 4/23/99 | 2 |
| 3 | Lobby Tableman | 15 | 4/7/99 | 3 |
| 4 | Betty Kezer | 31 | 4/15/99 | 4 |

TABLE 36: INITIAL PRODUCT DIMENSION TABLE

| key | Product_Name | Date | key_REAL |
|-----|--------------|---------|----------|
| 2 | Cool Stuff 2 | 4/30/99 | 2 |
| 3 | Wowzer Pro | 4/30/99 | 3 |
| 4 | Neato Lite | 4/30/99 | 4 |

The **key_REAL** column records the first row containing an entry for that **sskey**. In an initial load, the **key_REAL** value is always equal to the **key** value.

Each dimension table also has an **UNKNOWN** row, which always has a **key** value of 1. The **UNKNOWN** row is not illustrated in these tables.

The datamart also has mapping tables that map **sskeys** to dimension table **keys**. For example, the Customer mapping table might look like the Table 37.

TABLE 37: INITIAL CUSTOMER MAPPING TABLE

| iss | sskey | key |
|-----|-------|-----|
| 2 | bobb | 2 |
| 2 | lobt | 3 |
| 2 | betk | 4 |

In a subsequent extraction, you might load the following new data into the Customer staging table (Table 38).

TABLE 38: SECOND CUSTOMER STAGING TABLE

| ikey | sskey | iss | Customer_Name | Age | Date |
|------|-------|-----|----------------|-----|---------|
| 1 | jrd | 2 | J.R. Dobbs | 23 | 5/10/99 |
| 2 | lobt | 2 | Lobby Tableman | 16 | 5/5/99 |
| 3 | jrd | 2 | J.R. Dobbs | 20 | 5/2/99 |
| 4 | bobb | 2 | Bob Bobster | 47 | 4/23/99 |

This data can be merged into the dimension tables in different ways, depending on your choice of dimension semantic.

Note that all dimension semantics ignore the first row, since there is another staging table row with the same **sskey** and a higher **ikey** value. All semantics also ignore the last row, since it is a duplicate of a row that is already in the dimension table (row 2).

SLOWLY CHANGING DIMENSIONS

If the new data is merged with a Slowly Changing Dimensions semantic, then the revised dimension table looks like Table 39.

TABLE 39: CUSTOMER DIMENSION TABLE AFTER SLOWLY CHANGING DIMENSIONS SEMANTIC

| key | Customer_Name | Age | Date | key_REAL |
|-----|----------------|-----|---------|----------|
| 2 | Bob Bobster | 47 | 4/23/99 | 2 |
| 3 | Lobby Tableman | 15 | 4/7/99 | 3 |
| 4 | Betty Kezer | 31 | 4/15/99 | 4 |
| 5 | Lobby Tableman | 16 | 5/5/99 | 3 |
| 6 | J.R. Dobbs | 20 | 5/2/99 | 6 |

The new data for Lobby Tableman is appended to the dimension table. The **key_REAL** value of the new row points to the first row with that **sskey**, which is row 3.

The revised mapping table looks like Table 40.

TABLE 40: CUSTOMER MAPPING TABLE AFTER SLOWLY CHANGING DIMENSIONS SEMANTIC

| iss | sskey | key |
|-----|-------|-----|
| 2 | bobb | 2 |
| 2 | lobt | 5 |
| 2 | betk | 4 |
| 2 | jrd | 6 |

The mapping table now associates the **sskey** for Lobby Tableman with the new row, and all new facts that refer to customer **sskey lobt** will have a Customer foreign key of 5. However, previously extracted facts that referred to **sskey lobt** continue to have a Customer foreign key of 3.

APPENDIX F: SEMANTIC TYPES

FIRST DIMENSION VALUE

If the new data is merged with a First Dimension Value semantic, then the revised dimension table looks like Table 41.

TABLE 41: CUSTOMER DIMENSION TABLE AFTER FIRST DIMENSION VALUE SEMANTIC

| key | Customer_Name | Age | Date | key_REAL |
|-----|----------------|-----|---------|----------|
| 2 | Bob Bobster | 47 | 4/23/99 | 2 |
| 3 | Lobby Tableman | 15 | 4/7/99 | 3 |
| 4 | Betty Kezer | 31 | 4/15/99 | 4 |
| 5 | J.R. Dobbs | 20 | 5/2/99 | 5 |

The data for the new customer has been appended to the table, and the revised data for Lobby Tableman has been discarded.

The revised mapping table looks like Table 42

TABLE 42: CUSTOMER MAPPING TABLE AFTER FIRST DIMENSION VALUE SEMANTIC

| iss | sskey | key |
|-----|-------|-----|
| 2 | bobb | 2 |
| 2 | lobt | 3 |
| 2 | betk | 4 |
| 2 | jrd | 5 |

LATEST DIMENSION VALUE

If the new data is merged with a Latest Dimension Value semantic, then the revised dimension table looks like Table 43.

TABLE 43: CUSTOMER DIMENSION TABLE AFTER LATEST DIMENSION VALUE SEMANTIC

| key | Customer_Name | Age | Date | key_REAL |
|-----|----------------|-----|---------|----------|
| 2 | Bob Bobster | 47 | 4/23/99 | 2 |
| 3 | Lobby Tableman | 16 | 5/5/99 | 3 |
| 4 | Betty Kezer | 31 | 4/15/99 | 4 |
| 5 | J.R. Dobbs | 20 | 5/2/99 | 5 |

The data for the new customer has been appended to the table, and the revised data for Lobby Tableman has been used to update row 3. The revised mapping table looks like Table 44.

TABLE 44: CUSTOMER MAPPING TABLE AFTER LATEST DIMENSION VALUE SEMANTIC

| iss | sskey | key |
|-----|-------|-----|
| 2 | bobb | 2 |
| 2 | lobt | 3 |
| 2 | betk | 4 |
| 2 | jrd | 5 |

REWRITE DIMENSION

If the new data is merged with a Rewrite Dimension semantic, then the revised dimension table looks like Table 45.

TABLE 45: CUSTOMER DIMENSION TABLE AFTER REWRITE DIMENSION SEMANTIC

| key | Customer_Name | Age | Date | key_REAL |
|-----|----------------|-----|---------|----------|
| 2 | Bob Bobster | 47 | 4/23/99 | 2 |
| 3 | Lobby Tableman | 16 | 5/5/99 | 3 |
| 4 | Betty Kezer | 31 | 4/15/99 | 4 |
| 5 | Lobby Tableman | 16 | 5/5/99 | 3 |
| 6 | J.R. Dobbs | 20 | 5/2/99 | 6 |

As with the Slowly Changing Dimensions semantic, the revised data for Lobby Tableman is appended to the table. In addition, the old entry (in row 3) is updated with the new information.

The revised mapping table looks like Table 46.

TABLE 46: CUSTOMER MAPPING TABLE AFTER REWRITE DIMENSION SEMANTIC

| iss | sskey | key |
|-----|-------|-----|
| 2 | bobb | 2 |
| 2 | lobt | 5 |
| 2 | betk | 4 |
| 2 | jrd | 6 |

FACT SEMANTICS

After the initial extraction, you might have the following data in the fact-staging table (Table 47).

TABLE 47: INITIAL ORDER STAGING TABLE

| ikey | sskey | iss | Cust | Prod | Date | trans | proc | Qty |
|------|-------|-----|------|------|---------|-------|------|-----|
| 1 | 41N | 2 | lobt | neat | 4/22/99 | 1 | 1 | 4 |
| 2 | 65R | 2 | bobb | neat | 4/8/99 | 1 | 1 | 6 |
| 3 | 57Q | 2 | bobb | wow | 4/23/99 | 1 | 1 | 3 |
| 4 | 48Z | 2 | lobt | cool | 4/7/99 | 1 | 1 | 7 |
| 5 | 91B | 2 | lobt | wow | 4/1/99 | 1 | 1 | 1 |
| 6 | 46C | 2 | betk | cool | 4/19/99 | 1 | 1 | 2 |
| 7 | 55T | 2 | bobb | cool | 4/17/99 | 1 | 1 | 3 |

Here **Cust** is the Customer **sskey**, **Prod** is the Product **sskey**, **trans** is the transtype key, **proc** is the process key, and **Qty** is the quantity ordered.

When this data is loaded into the datamart with an initial load fact semantic, you get a datamart table like Table 48.

TABLE 48: INITIAL ORDER FACT TABLE

| seq | sskey | iss | Cust | Prod | Date | trans | Qty |
|-----|-------|-----|------|------|---------|-------|-----|
| 1 | 41N | 2 | 3 | 4 | 4/22/99 | 1 | 4 |
| 2 | 65R | 2 | 2 | 4 | 4/8/99 | 1 | 6 |
| 3 | 57Q | 2 | 2 | 3 | 4/23/99 | 1 | 3 |
| 4 | 48Z | 2 | 3 | 2 | 4/7/99 | 1 | 7 |
| 5 | 91B | 2 | 3 | 3 | 4/1/99 | 1 | 1 |
| 6 | 46C | 2 | 4 | 2 | 4/19/99 | 1 | 2 |
| 7 | 55T | 2 | 2 | 2 | 4/17/99 | 1 | 3 |

Here **Cust** and **Prod** are the integer foreign keys for the dimension table rows that correspond to a fact, and **seq** is a sequential numbering of the fact table rows.

In a subsequent extraction, you might load the following new data into the Order staging table (Table 49).

TABLE 49: NEW ORDER STAGING TABLE

| ikey | sskey | iss | Cust | Prod | Date | trans | proc | Qty |
|------|-------|-----|------|------|---------|-------|------|-----|
| 1 | 57Q | 2 | bobb | wow | 4/23/99 | 1 | 2 | 3 |
| 2 | 48Z | 2 | lobt | cool | 4/7/99 | 1 | 2 | 5 |
| 3 | 46C | 2 | betk | cool | 4/19/99 | 1 | 1 | 1 |
| 4 | 11L | 2 | betk | wow | 4/7/99 | 1 | 1 | 1 |
| 5 | 65R | 2 | bobb | neat | 4/8/99 | 1 | 2 | 7 |
| 6 | 71Z | 2 | jrd | neat | 5/8/99 | 1 | 2 | 5 |
| 7 | 41N | 2 | lobt | neat | 4/24/99 | 1 | 1 | 3 |
| 8 | 26F | 2 | jrd | wow | 5/14/99 | 1 | 1 | 10 |
| 9 | 57Q | 5 | jrd | cool | 4/20/99 | 1 | 1 | 3 |
| 10 | 55T | 2 | bobb | neat | 4/17/99 | 1 | 2 | 3 |

This data can be merged into the fact tables in different ways, depending on your choice of both fact and dimension semantics.

TRANSACTIONAL

If the new data is merged with a Transactional semantic, then all transactional facts (that is, facts with a **process_key** value of 1) are added to the fact table, unless the fact table already contains a fact with the same **sskey** and the same date or a later date. State-like facts (that is, facts with a **process_key** value of 2) are ignored.

The exact structure of the revised fact table depends on the semantic that was used to update the Customer dimension. If the Customer dimension is updated with a First Dimension Value or Latest Dimension Value semantic, then the fact table looks like Table 50.

TABLE 50: ORDER FACT TABLE, TRANSACTIONAL REVISION 1

| seq | sskey | iss | Cust | Prod | Date | trans | Qty |
|-----|-------|-----|------|------|---------|-------|-----|
| 1 | 41N | 2 | 3 | 4 | 4/22/99 | 1 | 4 |
| 2 | 65R | 2 | 2 | 4 | 4/8/99 | 1 | 6 |
| 3 | 57Q | 2 | 2 | 3 | 4/23/99 | 1 | 3 |
| 4 | 48Z | 2 | 3 | 2 | 4/7/99 | 1 | 7 |
| 5 | 91B | 2 | 3 | 3 | 4/1/99 | 1 | 1 |
| 6 | 46C | 2 | 4 | 2 | 4/19/99 | 1 | 2 |
| 7 | 55T | 2 | 2 | 2 | 4/17/99 | 1 | 3 |
| 8 | 11L | 2 | 4 | 3 | 4/1/99 | 1 | 1 |
| 9 | 41N | 2 | 3 | 4 | 4/24/99 | 1 | 3 |
| 10 | 26F | 2 | 5 | 3 | 5/14/99 | 1 | 10 |
| 11 | 57Q | 5 | 5 | 2 | 4/20/99 | 1 | 3 |

Note the following considerations:

- No staging table rows with a **process_key** value of 2 are added to the fact table.
- If a staging table row has the same **sskey** as a fact table row and the same date value as or an earlier date value than some row of the fact table, then that staging table row is not added to the fact table. For example, row 3 of the staging table was not added to the fact table because it has the same **sskey** and the same date as row 6 of the fact table.

- If a staging table row has the same **sskey** value but a different **iss** value, then it is treated as if it had a different **sskey** value. For example, row 9 of the staging table has the same **sskey** value as row 3 of the fact table, but a different **iss** value. Therefore, it is added to the fact table, even though it has an earlier date.
- All other staging table rows are added to the fact table.

If the Customer dimension is updated with a Slowly Changing Dimensions or Rewrite Dimension semantic, then the fact table looks like Table 51.

TABLE 51: ORDER FACT TABLE, TRANSACTIONAL REVISION 2

| seq | sskey | iss | Cust | Prod | Date | trans | Qty |
|-----|-------|-----|------|------|---------|-------|-----|
| 1 | 41N | 2 | 3 | 4 | 4/22/99 | 1 | 4 |
| 2 | 65R | 2 | 2 | 4 | 4/8/99 | 1 | 6 |
| ... | ... | ... | ... | ... | ... | ... | ... |
| 8 | 11L | 2 | 4 | 3 | 4/1/99 | 1 | 1 |
| 9 | 41N | 2 | 5 | 4 | 4/24/99 | 1 | 3 |
| 10 | 26F | 2 | 6 | 3 | 5/14/99 | 1 | 10 |
| 11 | 57Q | 5 | 6 | 2 | 4/20/99 | 1 | 3 |

The important difference here is that the new order for Lobby Tableman (row 7 of the staging table, row 9 of the fact table) is entered with a customer key value that points to the dimension table row with updated values (row 5).

The existing orders for Lobby Tableman still have customer keys that point to the dimension table row with old values (row 3).

TRANSACTIONAL/STATE-LIKE

If the new data is merged with a Transactional/State-Like semantic, then transactional facts are added as they are with a Transactional semantic type. State-like facts (that is, facts with a **process_key** value of 2) are assumed to reflect the current state of the fact quantities for a given **sskey**. If the fact table values do not match those in the state-like fact, then transactions are added to the fact table to adjust those values.

The structure of the revised fact table again depends on the semantic that was used to update the Customer dimension.

If the Customer dimension is updated with a First Dimension Value or Latest Dimension Value semantic, then the fact table looks like Table 52

TABLE 52: ORDER FACT TABLE, TRANSACTIONAL/STATE-LIKE REVISION 1

(1 OF 2)

| seq | sskey | iss | Cust | Prod | Date | trans | Qty |
|-----|-------|-----|------|------|---------|-------|-----|
| 1 | 41N | 2 | 3 | 4 | 4/22/99 | 1 | 4 |
| 2 | 65R | 2 | 2 | 4 | 4/8/99 | 1 | 6 |
| 3 | 57Q | 2 | 2 | 3 | 4/23/99 | 1 | 3 |
| 4 | 48Z | 2 | 3 | 2 | 4/7/99 | 1 | 7 |
| 5 | 91B | 2 | 3 | 3 | 4/1/99 | 1 | 1 |
| 6 | 46C | 2 | 4 | 2 | 4/19/99 | 1 | 2 |
| 7 | 55T | 2 | 2 | 2 | 4/17/99 | 1 | 3 |
| 8 | 48Z | 2 | 3 | 2 | 4/7/99 | 1 | -2 |
| 9 | 11L | 2 | 4 | 3 | 4/1/99 | 1 | 1 |
| 10 | 65R | 2 | 2 | 4 | 4/8/99 | 1 | 1 |
| 11 | 71Z | 2 | 5 | 4 | 5/8/99 | 1 | 5 |
| 12 | 41N | 2 | 3 | 4 | 4/24/99 | 1 | 3 |
| 13 | 26F | 2 | 5 | 3 | 5/14/99 | 1 | 10 |

TABLE 52: ORDER FACT TABLE, TRANSACTIONAL/STATE-LIKE REVISION 1

(2 OF 2)

| seq | sskey | iss | Cust | Prod | Date | trans | Qty |
|-----|-------|-----|------|------|---------|-------|-----|
| 14 | 57Q | 5 | 5 | 2 | 4/20/99 | 1 | 3 |
| 15 | 55T | 2 | 2 | 2 | 4/17/99 | 1 | -3 |
| 16 | 55T | 2 | 2 | 4 | 4/17/99 | 1 | 3 |

Note the following considerations:

- Staging table rows with a **process_key** value of 1 are added to the fact table as with the Transactional semantic type.
- If a staging table row with a **process_key** value of 2 has the same dimensionality (that is, the same dimension key values) and the same **sskey** value as an existing fact table row, but a different fact value, then a difference row is added to the fact table. For example, row 5 of the staging table has the same **sskey** and dimensionality values as row 2 of the fact table, but the order quantity is 7 rather than 6. Row 10 of the revised fact table therefore has an order for a single unit, so that the orders for **sskey** 65R add up to 7. Similarly, row 8 updates the value in row 4.
- If a staging table row with a **process_key** value of 2 has the same **sskey** value as an existing fact table row but different dimensionality, then the old order is debooked and the new order is booked. For example, row 10 of the staging table has the same **sskey** as row 7 of the fact table, but a different product dimension key. Therefore, row 15 of the revised fact table debooks the order with the old dimension values, and row 16 rebooks it with the new dimension values.
- If a staging table row with a **process_key** value of 2 has the same dimensionality, the same **sskey** value, and the same fact values as an existing fact table row, then that staging table row is ignored. For example, row 1 of the staging table duplicates row 3 of the fact table, so it is ignored.

If the Customer dimension is updated with a Slowly Changing Dimensions or Rewrite Dimension semantic, then the fact table looks like Table 53.

TABLE 53: ORDER FACT TABLE, TRANSACTIONAL/STATE-LIKE REVISION 2

| seq | sskey | iss | Cust | Prod | Date | trans | Qty |
|-----|------------|-----|------|------|---------------|-------|-----------|
| 1 | 41N | 2 | 3 | 4 | 4/22/99 | 1 | 4 |
| 2 | 65R | 2 | 2 | 4 | 4/8/99 | 1 | 6 |
| 3 | 57Q | 2 | 2 | 3 | 4/23/99 | 1 | 3 |
| 4 | 48Z | 2 | 3 | 2 | 4/7/99 | 1 | 7 |
| 5 | 91B | 2 | 3 | 3 | 4/1/99 | 1 | 1 |
| 6 | 46C | 2 | 4 | 2 | 4/19/99 | 1 | 2 |
| 7 | 55T | 2 | 2 | 2 | 4/17/99 | 1 | 3 |
| 8 | 48Z | 2 | 3 | 2 | 4/7/99 | 1 | -7 |
| 9 | 48Z | 2 | 5 | 2 | 4/7/99 | 1 | 5 |
| 10 | 11L | 2 | 4 | 3 | 4/1/99 | 1 | 1 |
| ... | ... | ... | ... | ... | ... | ... | ... |
| 16 | 55T | 2 | 2 | 2 | 4/17/99 | 1 | -3 |
| 17 | 55T | 2 | 2 | 4 | 4/17/99 | 1 | 3 |

The difference here is in rows 8 and 9. Row 2 of the staging table refers to the same customer as row 4 of the fact table. However, since the data for Lobby Tableman has changed and the dimension has been updated with a Slowly Changing Dimensions or Rewrite Dimension semantic, a new row with revised data has been added to the dimension table. Since the state-like fact refers to a different dimension table row than that referred to by the initial transactional fact, it is treated as a state-like fact with changed dimensionality. Therefore, the original order is debooked and the new order is rebooked.

TRANSACTIONAL/STATE-LIKE/FORCE CLOSE

If the new data is merged with a Transactional/State-Like/Force Close semantic, then new data is merged in the same way as with Transactional/State-Like. In addition, the Force Close logic is applied to facts with **sskeys** that do not appear in the staging table. When a fact table **sskey** does not appear in the staging table, and there is at least one row in the fact table with that **sskey** and an Order transaction type (that is, with a **transtype_key** value in the range 1-99), then that **sskey** is forced closed. For such an **sskey**, all facts with the **sskey** (regardless of **transtype**) are added. If this sum is not zero, then a new fact that forces the sum to be zero is added to the fact table. Note that if all fact table rows with a given **sskey** value have **transtype_key** values greater than 99, then that fact is not affected by the force close logic.

The Transactional/State-Like/Force Close semantic type is most commonly used with the pseudo-order approach described on page 457, but it can be applied to any fact and staging tables. When applied to the example fact and staging tables in this section, the exact structure of the revised fact table again depends on the semantic that was used to update the Customer dimension.

If the Customer dimension is updated with a First Dimension Value or Latest Dimension Value semantic, then the fact table looks like Table 54.

TABLE 54: ORDER FACT TABLE, TRANSACTIONAL/STATE-LIKE/FORCE CLOSE REVISION 1

| seq | sskey | iss | Cust | Prod | Date | trans | Qty |
|-----|-------|-----|------|------|---------|-------|-----|
| 1 | 41N | 2 | 3 | 4 | 4/22/99 | 1 | 4 |
| ... | ... | ... | ... | ... | ... | ... | ... |
| 5 | 91B | 2 | 3 | 3 | 4/1/99 | 1 | 1 |
| 6 | 46C | 2 | 4 | 2 | 4/19/99 | 1 | 2 |
| 7 | 55T | 2 | 2 | 2 | 4/17/99 | 1 | 3 |
| ... | ... | ... | ... | ... | ... | ... | ... |
| 15 | 55T | 2 | 2 | 2 | 4/17/99 | 1 | -3 |
| 16 | 55T | 2 | 2 | 4 | 4/17/99 | 1 | 3 |
| 17 | 91B | 2 | 3 | 4 | 4/1/99 | 1 | -1 |

Note the following considerations:

- Since there is no row in the staging table with **sskey 91B**, row 17 of the revised fact table de-books the order in row 5.
- The order in row 6 is not de-booked, because row 3 of the staging table has the same **sskey**.
- All staging table rows are treated as they would be with a Transactional/State-Like semantic type. In particular, row 3 of the staging table is not added to the fact table, because its date is before that of row 6 of the fact table.

If the Customer dimension is updated with a Slowly Changing Dimensions or Rewrite Dimension semantic, then the fact table looks like Table 55.

TABLE 55: ORDER FACT TABLE, TRANSACTIONAL/STATE-LIKE/FORCE CLOSE REVISION 2

| seq | sskey | iss | Cust | Prod | Date | trans | Qty |
|-----|-------|-----|------|------|---------|-------|-----|
| 1 | 41N | 2 | 3 | 4 | 4/22/99 | 1 | 4 |
| ... | ... | ... | ... | ... | ... | ... | ... |
| 7 | 55T | 2 | 2 | 2 | 4/17/99 | 1 | 3 |
| 8 | 48Z | 2 | 3 | 2 | 4/7/99 | 1 | -7 |
| 9 | 48Z | 2 | 5 | 2 | 4/7/99 | 1 | 5 |
| 10 | 11L | 2 | 4 | 3 | 4/1/99 | 1 | 1 |
| ... | ... | ... | ... | ... | ... | ... | ... |
| 16 | 55T | 2 | 2 | 2 | 4/17/99 | 1 | -3 |
| 17 | 55T | 2 | 2 | 4 | 4/17/99 | 1 | 3 |
| 18 | 91B | 2 | 3 | 3 | 4/1/99 | 1 | -1 |

The only difference here is that rows 8 and 9 de-book and re-book the order with sskey 48Z, as with the Transactional/State-Like semantic type.

RELOAD DATE FACT

If the new data is merged with a Reload Date Fact semantic, then all facts in the staging table are placed into a new fact table. In addition, all facts from the old fact table that have a date that is earlier than the earliest date in the new fact table are appended to the new fact table.

The exact structure of the revised fact table again depends on the semantic that was used to update the Customer dimension.

If the Customer dimension is updated with a First Dimension Value or Latest Dimension Value semantic, then the fact table looks like Table 56.

TABLE 56: ORDER FACT TABLE, RELOAD DATE FACT REVISION 1

| seq | sskey | iss | Cust | Prod | Date | trans | Qty |
|-----|-------|-----|------|------|---------|-------|-----|
| 1 | 57Q | 2 | 2 | 3 | 4/23/99 | 1 | 3 |
| 2 | 48Z | 2 | 3 | 2 | 4/7/99 | 1 | 5 |
| 3 | 46C | 2 | 4 | 2 | 4/19/99 | 1 | 1 |
| 4 | 11L | 2 | 4 | 3 | 4/7/99 | 1 | 1 |
| 5 | 65R | 2 | 2 | 4 | 4/8/99 | 1 | 7 |
| 6 | 71Z | 2 | 5 | 4 | 5/8/99 | 1 | 5 |
| 7 | 41N | 2 | 3 | 4 | 4/24/99 | 1 | 3 |
| 8 | 26F | 2 | 5 | 3 | 5/14/99 | 1 | 10 |
| 9 | 57Q | 5 | 5 | 2 | 4/20/99 | 1 | 3 |
| 10 | 55T | 2 | 2 | 4 | 4/17/99 | 1 | 3 |
| 11 | 91B | 2 | 3 | 3 | 4/1/99 | 1 | 1 |

This is essentially the staging table with older fact-table data appended.

If the Customer dimension is updated with a Slowly Changing Dimensions or Rewrite Dimension semantic, then the fact table looks like Table 57.

TABLE 57: ORDER FACT TABLE, RELOAD DATE FACT REVISION 2

| seq | sskey | iss | Cust | Prod | Date | trans | Qty |
|-----|-------|-----|------|------|---------|-------|-----|
| 1 | 57Q | 2 | 2 | 3 | 4/23/99 | 1 | 3 |
| 2 | 48Z | 2 | 5 | 2 | 4/7/99 | 1 | 5 |
| 3 | 46C | 2 | 4 | 2 | 4/19/99 | 1 | 1 |
| 4 | 11L | 2 | 4 | 3 | 4/7/99 | 1 | 1 |
| 5 | 65R | 2 | 2 | 4 | 4/8/99 | 1 | 7 |
| 6 | 71Z | 2 | 6 | 4 | 5/8/99 | 1 | 5 |
| 7 | 41N | 2 | 5 | 4 | 4/24/99 | 1 | 3 |
| 8 | 26F | 2 | 6 | 3 | 5/14/99 | 1 | 10 |
| 9 | 57Q | 5 | 6 | 2 | 4/20/99 | 1 | 3 |
| 10 | 55T | 2 | 2 | 4 | 4/17/99 | 1 | 3 |
| 11 | 91B | 2 | 3 | 3 | 4/1/99 | 1 | 1 |

The only difference here is in the dimension keys for the new values. Notice that the new entries for Lobby Tableman have key 5, while the entry (in row 11) that was imported from the old fact table still has a key value of 3.

JOB TYPES AND SEMANTICS

The choice of job type limits the semantics that can be used. The permitted semantics for each job type are listed below.

INCREMENTAL JOBS

The permitted dimension semantics for an Incremental job are:

- Slowly Changing Dimension
- First Dimension Value
- Backfeed Dimension

The permitted fact semantics for an Incremental job are:

- Transactional
- Transactional/State-Like
- Transactional/State-Like/Force Close
- Pipelined
- Backfeed Fact
- Count Unjoined

INITIAL JOBS

The permitted dimension semantics for an Initial job are:

- Initial Load Dimension
- Backfeed Dimension

The permitted fact semantics for an Initial job are:

- Initial Load Fact
- Reload Date Fact
- First/Last Fact

- Backfeed Fact
- Count Unjoined

MERGE JOBS

The permitted dimension semantics for a Merge job are:

- Slowly Changing Dimension
- First Dimension Value
- Backfeed Dimension

The permitted fact semantics for a Merge job are:

- Transactional
- Transactional/State-Like
- Transactional/State-Like/Force Close
- Pipelined
- Initial Load Fact
- Reload Date Fact
- First/Last Fact
- Backfeed Fact
- Count Unjoined

REBUILD JOBS

The permitted dimension semantics for a Rebuild job are:

- Slowly Changing Dimension
- First Dimension Value
- Latest Dimension Value
- Rewrite Dimension

- Initial Load Dimension
- Backfeed Dimension

The permitted fact semantics for a Rebuild job are:

- Transactional
- Transactional/State-Like
- Transactional/State-Like/Force Close
- Pipelined
- Initial Load Fact
- Reload Date Fact
- First/Last Fact
- Backfeed Fact
- Count Unjoined

NON-PARTITIONED JOBS

The permitted dimension semantics for a Non-Partitioned job are:

- Slowly Changing Dimension
- First Dimension Value
- Latest Dimension Value
- Rewrite Dimension
- Initial Load Dimension
- Backfeed Dimension

The permitted fact semantics for a Non-Partitioned job are:

- Transactional
- Transactional/State-Like
- Transactional/State-Like/Force Close
- Pipelined

- Initial Load Fact A/B
- Reload Date Fact A/B
- Backfeed Fact
- Count Unjoined

APPENDIX F: SEMANTIC TYPES